1

SOLUTION OF POTENTIAL FLOW PAST AN ELASTIC BODY USING THE BOUNDARY ELEMENT TECHNIQUE

THESIS

Norma F. Taylor

AFIT/GAE/AA/88D-37

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

(1)

AFIT/GAE/AA/88D-37

SOLUTION OF POTENTIAL FLOW PAST AN ELASTIC BODY USING THE BOUNDARY ELEMENT TECHNIQUE

THESIS

Norma F. Taylor

AFIT/GAE/AA/88D-37

AFIT/GAE/AA/88D-37

SOLUTION OF POTENTIAL FLOW PAST AN ELASTIC BODY USING THE BOUNDARY ELEMENT TECHNIQUE

THESIS

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University
In Partial Fulfillment of the
Requirements for the Degree of
Master of Science in Aerospace Engineering

Norma F. Taylor

December 1988

Acknowledgements

Many people have supported me throughout the course of this research, and I owe thanks to them all. First and foremost is my thesis advisor, Major Hudson. When the effort seemed overwhelming, his advice and encouragement would keep me going. I would also like to thank my co-workers and supervisors at the 4950th Test Wing for their patience during the many times I was away from the office throughout the past year. Finally, thanks to all my family and friends who were always willing to listen to my tales of woe and then convince me it would be worth all the troubles in the end.

Norma F. Taylor

Table of Contents

Page

Acknowledgements ii

List of Figures v

List of Symbols vii

Abstract ix

I. Introduction 1

II. Theoretical Background 4

Numerical Approximation Techniques 4
Boundary Element Formulation 7
Direct and Indirect Methods 9
Elements 10
Internal and External Domains 11

III. Development of Models 13

Potential Flow 13
Velocity and Pressure Distribution 19
Elastic Structure 20
Assembly of Subregions 23

IV. Implementation 26

Geometry 26
Potential Flow 28
Velocity and Pressure Field 30
Elastic Deformations 34
Solution of Linear Equations 34
Continuity Considerations 34
Iterative Process 35

V. Discussion and Results 37

Effect of Element Size on Solution 37
Effect of Approximation Order on Solution 38
Effect of End Node Location on Solution 38
Comparison of Potential Flow Results With Theory 39
Effect of Number of Elements And Approximating Order On Solution Time 39
Elastic Deformation Results 40
Rigid-Body Constraint 41
Comparison With Finite Element Model 41
Iteration of Fluid/Structure Solutions 41

VI. Conclusions and Recommendations 43

Page

Appendix A: Investigation Results 45

Appendix B: Program Routines 67

Appendix C: Background Concepts 112

Bibliography 115

Vita 116

List of Figures

Figure Page

1. Domain and Boundary Notation 6

2. Applying the BEM Equation to the Boundary 9

3. Geometry Elements 10

4. Approximating Order of Functions and Corresponding Nodes 11

5. Internal and External Problem Definitions 12

6. Source and Field Points 14

7. Elemental Coordinate System 15

8. Elemental and Global Node Numbering 17

9. Structure Geometry 21

10. Global Node Numbering for Structure 25

11. Geometry Modeled With 8 Flat Elements 27

12. Nodal Spacing Within an Element 28

13. Singular Panel Approximation Definitions 31

14. Definition of Angle Around Cylinder 38

15. Direction of Deformations on the Structure 40

A1. Effect of Element Size on Solution, Constant Modeling of Cp 46

A2. Effect of Element Size on Solution, Linear Modeling of Cp 47

A3. Effect of Element Size on Solution, Parabolic Modeling of Cp 48

A4. Effect of Function Approximation on Solution, 8 Elements 49

A5. Effect of Function Approximation on Solution, 40 Elements 50

A6. Effect of Function Approximation on Solution, 120 Elements 51

A7. Effect of End Node Location on Solution, 40 Elements, Parabolic Approximation of Cp 52

A8. Effect of End Node Location on Solution, 40 Elements, Parabolic Approximation of Cp (Magnified)53

A9. Comparison of X Component of Velocity With Theory, 120 Elements, Linear Approximation . . 54

A10. Comparison of Y Component of Velocity With Theory, 120 Elements, Linear Approximation . 55

A11. Comparison of Velocity With Theory, 120 Elements, Linear Approximation 56

A12. Comparison of Coefficient of Pressure With Theory, 120 Elements, Linear Approximation . . . 57

A13. Comparison of Potential With Theory, 120 Elements, Linear Approximation 58

Figure Page

A14. X Component of Deformation, 40 Elements, Linear Approximation 59

A15. Y Component of Deformation, 40 Elements, Linear Approximation 60

A16. X Component of Deformation From FEM, 40 Elements, Linear Approximation 61

A17. Y Component of Deformation From FEM, 40 Elements, Linear Approximation 62

A18. X Component of Deformation After 1 Pass, 8 Elements, Linear Approximation 63

A19. Iterative X Component of Deformation, 3-9 Iterations, 8 Elements, Linear Approximation 64

A20. Y Component of Deformation After 1 Pass, 8 Elements, Linear Approximation 65

A21. Iterative Y Component of Deformation, 3-9 Iterations, 8 Elements, Linear Approximation 66

List of Symbols

Symbol	Definition
b_j	Body force in j direction
c^i	Coefficient related to boundary geometry
Cp	Coefficient of pressure
e	Radius of region located about point i
$\|G\|$	Jacobian
G	Young's modulus of elasticity
i	Nodal point at which collocation is performed
j	Total number of elements around the boundary
m	Total number of nodes around the boundary
n_i	Component of outward normal in i direction
N_i	Shape function for node i on element
P	Pressure
Px, Py	Components of pressure in x and y direction
q	du/dn
q*	dw*/dn
r	Order of approximation function and number of nodes on element
r	Distance between source point and field point
u(x)	Unknown function
u	Component of perturbation velocity in x direction
u_x	Deformation in x direction
u_y	Deformation in y direction
v	Component of perturbation velocity in y direction
V_p	Total perturbation velocity
w*(x,x)	Free-space Green's function, also the weighted residual weighting function
x	Field point where integration is performed
x_i	X coordinate of point i
y_i	Y coordinate of point i
δ_{ij}	Kronecker delta
δ	Distance from end node point to element corner

Symbol	Definition
Δ^i	Direc delta function at point i
Γ	Boundary of domain
η	Elemental coordinate
ν	Poisson's ratio
θ	Angle between element and x-axis
Ω	Domain under consideration
ξ	Source point

Abstract

This thesis describes the development of a Fortran computer code which models the interaction between an incompressible, potential flow and a homogeneous, elastic structure. The boundary element technique was chosen over finite differences and finite elements because of its ability to numerically approximate both the fluid and structural behavior with a common definition of the fluid/structure boundary.

The ability to accurately model solid and fluid boundaries can be quite important in the fields of aeroelasticity and structural analysis. The nature of these boundaries is what determines the final solution to a problem of fluid flow past an elastic body. Often the complexity of defining and tracking the boundary and its associated boundary conditions has led the user to assumptions of rigid bodies, and therefore rigid boundaries. Certainly the tasks of defining the domain grids for finite difference and finite element techniques have not simplified the process.

In the computer code developed for this thesis the fluid and structural governing equations are simultaneously solved to determine the pressure about the structure and the corresponding elastic deformations. The deformations are applied to the original boundary, resulting in a new geometry. This new geometry is used to recalculate the pressure field about the structure, and the process is iterated until a final steady-state solution is obtained.

While simplifying assumptions have been made in the execution of this thesis, the general boundary element techique has the ability to model complex, higher order problems. The initial results obtained during the course of this work show promise, and follow-on studies are recommended.

SOLUTION OF POTENTIAL FLOW PAST AN ELASTIC BODY USING THE BOUNDARY ELEMENT TECHNIQUE

I. Introduction

This thesis investigates the simultaneous solution of the interaction between an incompressible, potential flow and a homogeneous, elastic structure using the boundary element technique. The ability to accurately model the fluid and structural boundaries can be quite important in the fields of aeroelasticity as the emphasis in aeroelasticity is;

> . . . phenomena which exhibit appreciable reciprocal interaction (static or dynamic) between aerodynamic forces and the deformations induced thereby in the structure of a flying vehicle, its control mechanisms, or its propulsion system. (1:1)

For example, the twist angle-of-attack of a wing is composed of a series of incremental twists each caused by the aerodynamic loading of the previous geometry (1: Eq 6-7). The original wing geometry causes aerodynamic loading which in turn produces a twist, or deformation, to the wing. This deformation causes a slightly changed loading which in turn causes additional twist, and so on. The ability to compute this incremental deformation determines the final solution to a problem of fluid flow past an elastic body. The traditional approach to aeroelasticity is to perform a modal analysis of the structure assuming structural shapes and determining the aerodynamic loads due to those shapes. The approach taken in this research presents a very different one in that the fluid and structure governing equations are simultaneously solved by iterating and converging on a final solution in the sense of the example just quoted. The iterative approach performed in this thesis is valid for steady-state problems, however the general boundary element technique has the capability to model non-steady problems. The problem might just as easily been formulated as the long-term response of a non-steady fluid/structure system.

The major thrust of this thesis is to research the application of the boundary element technique to compute the incremental deformation of an elastic structure in fluid flow. The boundary element, or boundary integral, method has been used to analyze the structure - fluid interaction problems in water waves. As described in References 6 and 11 a semi-infinite (half) domain was formed using the surface of the ocean as a free boundary, and the ocean bottom as a solid boundary. The solution was simultaneously solved using an unsteady approach with frequency domain decomposition.

The boundary element technique was chosen over finite differences and finite elements because of its ability to numerically approximate both the fluid and elastic structure governing equations with a common definition of the fluid/structure boundary.

In the iterative process the flow conditions (velocity and density of the fluid at infinity), the structural conditions (Young's modulus and Poisson's ratio), and the original boundary geometry specify the critical state. The first step of the process is to compute the fluid pressure field about the structure, and then determine the structural deformations caused by the pressure distribution. The new boundary geometry is automatically calculated from the deformations, and the pressure field recalculated. This process can be iterated to show how the structure changes shape.

An application of this program technique can be seen in a flight test program currently being performed at the 4950th Test Wing, Wright-Patterson Air Force Base. The MILSTAR airworthiness program is a flight test program to certify the airworthiness of a radome to cover a satellite antenna for the MILSTAR terminal system (4). Because of a clearance of only one-half inch between the antenna and the radome, one objective of the flight test program was to measure radome deformations to verify adequate clearance. The analytical prediction of the radome deformations became a major task of the test team, and was found to be highly manpower intensive. The aircraft/radome geometry was first entered into a computational fluid dynamics program called QUADPAN (10) which calculated the pressures about the radome assuming subsonic steady-state flow. QUADPAN itself is a type of boundary element code. The radome structure was also modeled with a NASTRAN finite element geometry grid. Both of these tasks took several weeks to set up. The users then ran the QUADPAN code at the desired flight conditions, fed the resulting pressures through a program to translate them from the center of the QUADPAN panels to the NASTRAN nodes, then input the translated pressure into the NASTRAN code to result in the final radome deformation. Information on the changes to the pressure field due to the deformations, and the resulting change in deformation would require extensive remodeling to the QUADPAN and NASTRAN geometry models after each iteration. Due to the extensive effort involved in this remodeling, no iterations were performed on the pressure/deformation solutions. Deformations recorded during flight were used to determine the accuracy of the predicted deformations.

The boundary element code presented in this thesis has several advantages to the above method. The users would only be required to set up the original aircraft and radome boundaries which is quite similar in work effort to the generation of the QUADPAN model. Once the flow conditions and structural properties were input, the program would then calculate the radome deformations and iterate to a steady-state solution. A more extensive unsteady boundary element program than that presented in this research could conceivably predict radome flutter using a time-marching scheme.

Subsequent chapters of this thesis covers a brief theoretical background of the boundary element models used in the computer program, a description of the implementation of the theory, results of the investigation, and conclusions and recommendations drawn from the research performed. While simplifying assumptions have been made in the execution of this investigation, the general boundary element technique has the ability to model complex governing equations and higher order approximation techniques. The initial results obtained during the course of this work show promise towards the suitability of the boundary element method for such as task, and follow-on studies are recommended to further develop a code usable in the field.

II. Theoretical Background

Numerical Approximation Techniques (2: Chap 2, 3: Chap 1)

During the recent decades the emphasis of engineering computations has shifted from analytical approximating techniques to massive numerical approximations. Some of the most popular numerical approximation techniques today are the finite difference method (FDM), the finite element method (FEM) and the boundary element method (BEM).

The first two methods are 'domain' methods.

> These techniques discretize the domain of the problem under consideration into a number of elements or cells. The governing equations of the problem are then approximated over the region by functions which fully or partially satisfy the boundary conditions. (2:1)

In the finite difference method a series of nodes in a domain grid is defined and the discretized version of the governing equation is satisfied only at those nodes. In the finite element method, the governing equation is satisfied in an average sense over a region or element. In the case of the FEM the integrations are conducted over the domain. In both methods, the user must discretize the domain as well as the boundary of the region under consideration.

The third method is known as the boundary integral method, or the boundary element method, and satisfies the governing equations throughout the domain but only approximates the boundary conditions. One of the first engineering applications of the BEM was the solution for Laplace-type problems by O.D. Kellogg in 1953 (7). J.L. Hess was also an important contributor to the development and use of boundary element methods in fluid mechanics problems (5). Integral equation techniques have been in development since the early 1950s, but they were overshadowed by the more popular finite difference and finite element techniques due to the difficulty of defining the appropriate Green's functions for the BEM. Recently, engineers have begun to rediscover boundary element methods and their advantages. A few of these advantages are;

a. The governing equations are reduced from domain integrals to boundary integrals thereby reducing the order of the problem by one spatial dimension.

b. Boundary method techniques can be coupled with other techniques to improve the accuracy of the solution.

c. The boundary method can simply model problems with infinite domains.

d. Finally, and most important to the application of this thesis, only the boundary of the domain must be discretized. Once the boundary geometry has been defined, both interior and exterior problems may be solved without discretizing either domain. This advantage facilitates the modeling and tracking of moving boundaries.

For practical engineering problems (full aircraft, three-dimensional geometry) current computational ability favors a mixture of boundary integral formulations where applicable, augmented by finite difference methods for regions where non-linear effects dominate.

All three of the approximation techniques are numerically related, and may be derived through the use of the weighted residual technique. As an example of the weighted residual formulation consider a potential function u on a domain Ω which satisfies the governing equation:

$$\nabla^2 u - b = 0 \quad \textit{in domain } \Omega \tag{1}$$

The boundary conditions are of two types

(a) Essential or Dirichlet conditions, such as $u = \bar{u}$ on Γ_1 (2)

(b) Natural or Neumann conditions, such as $\frac{\partial u}{\partial n} = \overline{\frac{\partial u}{\partial n}}$ on Γ_2 (3)

The barred terms are the known conditions. The entire boundary is composed of Γ_1 and Γ_2. The outward normal is defined as n (Figure 1).

When the function u and its boundary conditions are numerically approximated, errors are introduced into the problem solution. A weighting function w* with continuous first derivatives is used to distribute the error over the domain and boundary.

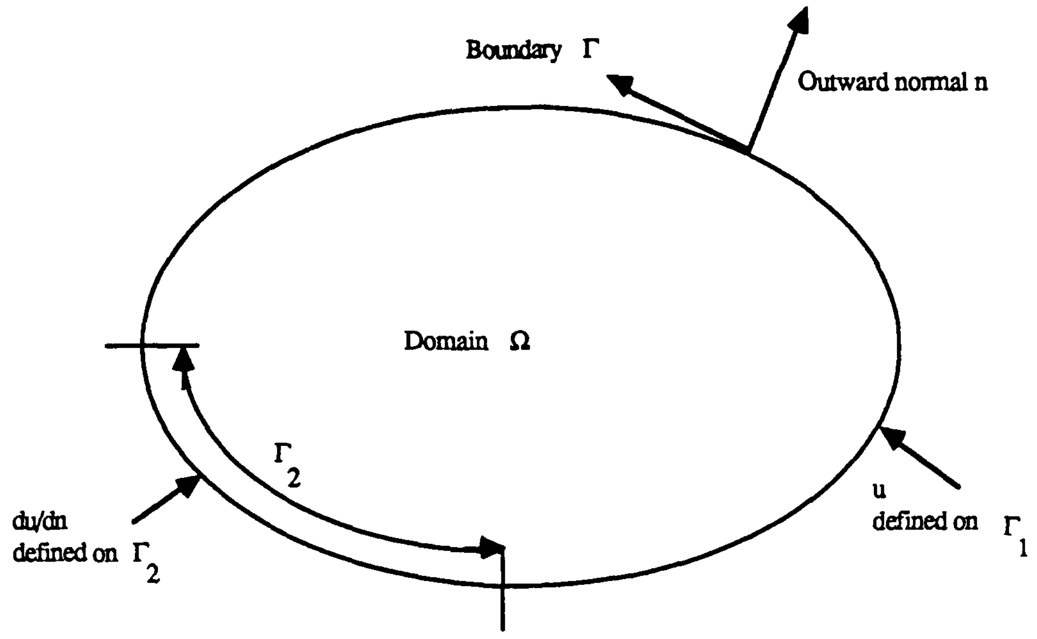


Figure 1. Domain and Boundary Notation

The weighted residual statement becomes;

$$\int_{\Omega} (\nabla^2 u - b)\, w^{*}\, d\Omega = 0 \quad \text{in domain } \Omega \tag{4}$$

Further discussion on the weighted residual technique is found in Appendix C. The finite difference method can be obtained from Equation 4, and is called the "original statement";

$$\int_{\Omega} \left(\nabla^2 u - b\right) w^{*}\, d\Omega = \int_{\Gamma} \frac{\partial u}{\partial n} w^{*}\, d\Gamma - \int_{\Gamma} u \frac{\partial w^{*}}{\partial n}\, d\Gamma \tag{5}$$

For finite differences, the basis functions u and w* are normally different, where w* is usually the Dirac delta function and the derivatives of u are local expansions such as Taylor series.

Integrating Equation 4 by parts once results in the "weak statement";

$$\int_\Omega \left(\frac{\partial u}{\partial x_k}\frac{\partial w^*}{\partial x_k} + bw^* \right) d\Omega = \int_\Gamma \frac{\partial u}{\partial n} w^* d\Gamma + \int_\Gamma u \frac{\partial w^*}{\partial n} d\Gamma \qquad (6)$$

Finite element techniques base numerical approximations on this form of the governing equation. Most finite element models are based on the method of Galerkin where both u and w* are related so that w* is a variation of u associated with virtual displacements or velocities.

Integrating Equation 6 by parts once more obtains the "inverse statement";

$$\int_\Omega (\nabla^2 w^*) u \, d\Omega = \int_\Gamma u \frac{\partial w^*}{\partial n} d\Gamma - \int_\Gamma w^* \frac{\partial u}{\partial n} d\Gamma + \int_\Omega b w^* d\Omega \qquad (7)$$

Equation 7 can be rewritten as;

$$\int_\Omega (\nabla^2 w^*) u \, d\Omega = \int_{\Gamma_1} \bar{u} \frac{\partial w^*}{\partial n} d\Gamma + \int_{\Gamma_2} u \frac{\partial w^*}{\partial n} d\Gamma$$

$$- \int_{\Gamma_1} \frac{\partial u}{\partial n} w^* d\Gamma - \int_{\Gamma_2} \overline{\frac{\partial u}{\partial n}} w^* d\Gamma + \int_\Omega b w^* d\Omega \qquad (8)$$

Equation 8 is the basis for the boundary element method. Generally the weighting function is chosen so that the left-side of the equation vanishes leaving the boundary terms.

Boundary Element Formulation (2: Chap 3, 3: Section 2.4)

If the boundary conditions are now defined as:

$$q = \frac{\partial u}{\partial n} \quad \text{and} \quad q^* = \frac{\partial w^*}{\partial n} \qquad (9)$$

the boundary element formulation is:

$$\int_{\Omega}\left(\nabla^{2} w^{*}\right) u \, d\Omega = -\int_{\Gamma_2} \bar{q} w^{*} d\Gamma - \int_{\Gamma_1} q\, w^{*} d\Gamma + \int_{\Gamma_2} u\, q^{*} d\Gamma + \int_{\Gamma_1} \bar{u}\, q^{*} d\Gamma + \int_{\Omega} b\, w^{*} d\Omega \qquad (10)$$

Assume that a concentrated charge is acting at point 'i'. The governing equation becomes:

$$\nabla^{2} u + \Delta^{i} = 0 \qquad (11)$$

where Δ^{i} is a Dirac delta function at point 'i'. The solution to this equation is called the fundamental solution where the effects of boundaries at infinity are considered. If equation 11 is satisfied with the fundamental solution then:

$$\int_{\Omega} u\,(\nabla^{2} w^{*})\, d\Omega = u^{i} \qquad (12)$$

Equation 10 now becomes:

$$u^{i} + \int_{\Gamma_2} u\, q^{*} d\Gamma + \int_{\Gamma_1} \bar{u}\, q^{*} d\Gamma = \int_{\Gamma_2} \bar{q}\, u^{*} d\Gamma + \int_{\Gamma_1} q\, u^{*} d\Gamma + \int_{\Omega} b\, w^{*} d\Omega \qquad (13)$$

Equation 13 is valid for any point in domain Ω, but must be taken to the boundary to solve for the boundary conditions. To do this a region with radius e is formed around the boundary point i (Figure 2). Equation 13 is separated into integrals around Γ_e and around Γ-Γ_e. By taking the limit as e approaches zero, the integrals over Γ - Γ_e are continuous. The integral over Γ_e for the right hand side contains a weak singularity but remains continuous. The left side integral contains a higher order singularity because of the derivative term. As the point i approaches the boundary along the normal, it produces a discontinuity as the point passes through the boundary.

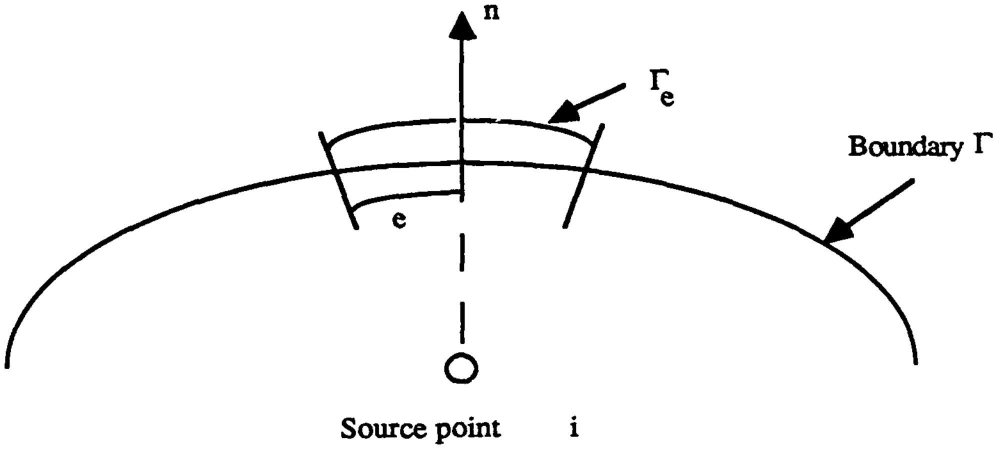


Figure 2. Applying the BEM Equation to the Boundary

On the boundary Equation 13 is reduced to:

$$2\,\pi\, c^i u^i + \int_{\Gamma_2} u\, q^* \, d\Gamma + \int_{\Gamma_1} \bar{u}\, q^* \, d\Gamma = \int_{\Gamma_2} \bar{q}\, u^* \, d\Gamma + \int_{\Gamma_1} q\, u^* \, d\Gamma + \int_{\Omega} b\, w^* \, d\Omega \qquad (14)$$

The term ($2\,\pi\, c^i$) represents the discontinuity and is a function of the boundary shape. By employing a method similar to that for evaluating the discontinuity, it can be shown that the term equals the internal angle of the boundary at the point i (3: 63,64). For a smooth boundary the term equals π and c^i becomes 1/2. A second method of calculating c^i will be discussed in Chapter 3.

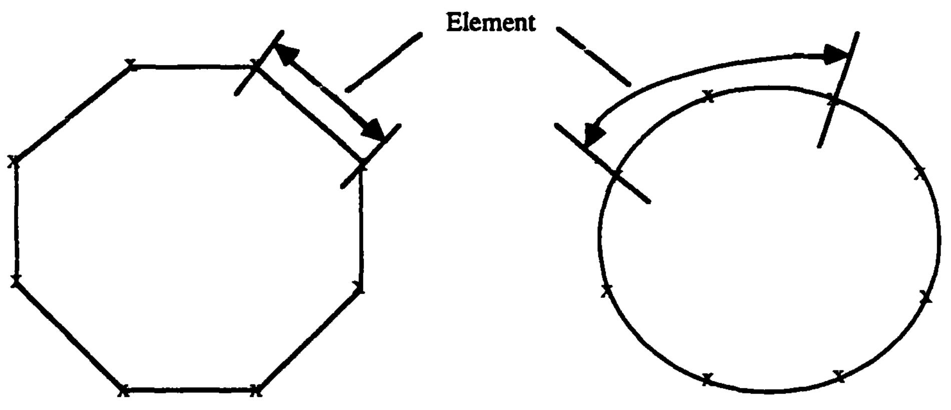
Direct and Indirect Methods

The direct formulation of the BEM is the formulation discussed above. It may be derived through the use of weighted residuals as shown, or through Green's third identity, Betti's or similar theorys. The final solution is the function u and its derivatives. For example, when modeling the elastic structure the final solution is the deformation at the node points. An advantage of the direct formulation is that a general surface with corners or edges may be used.

The indirect formulation of the Laplace equation problem sets up the problem as solely a single-layer potential or solely a double-layer potential of continuous source distributions over the boundary. The indirect method carries a restriction that the boundary must be smooth. The equations are then solved for the source densities. Further discussion on the indirect method may be found in Reference 7.

Elements

As the name 'boundary element method' implies, the boundary of domain Ω is divided into segments called elements (Figure 3). The general theory places no restrictions on the modeling of the element. For simplicity, linear (flat) elements were used in the development of this thesis, but higher order elements may be used to model the boundary more smoothly.

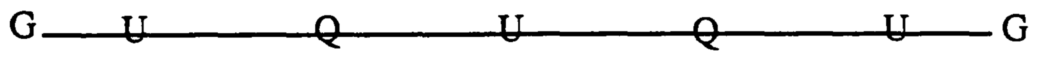


a) 8 linear elements (8 nodes)   b) 4 quadratic elements (8 nodes)

Figure 3. Geometry Elements

The order of the geometry model will determine the continuity of the geometry slope. Throughout a continuous boundary the du/dn terms may be calculated everywhere, but on a discontinuous boundary the du/dn terms are undefined at the ends of the elements because the normal is undefined. A discontinuous geometry slope will also cause the solution to display sharply singular behavior in the vicinity of the corners. The major disadvantage to continuous geometry models is the degree of effort involved in the implementation. For the purposes of this investigation, linear geometry modeling was used and therefore the geometry slope is discontinuous at the element corners. The consequences of using discontinuous geometry are discussed in Chapters 4 and 5

Approximations of the function u and its derivatives are assumed to vary over the element. The governing equation is then written at various locations on each element called nodes. The number of nodes on each element corresponds to the order of the approximating functions of u and q, and not necessarily to the order of the element geometry. For example, u may be modeled with a parabolic function , and therefore three function nodes, while the geometry element may be linear (Figure 4). The functions u and q do not have to be modeled with the same order, in fact it may be more advantageous to take q of one order less than u because it is the derivative of u. For the purpose of this research, the approximations of u and q were equivalent order for simplicity of implementation.



Order of Geometry = 1st (2 nodes)

Order of Function u = 2nd (3 nodes)

Order of Function q = 1st (2 nodes)

Figure 4. Approximating Order of Functions and Corresponding Nodes

Internal and External Domains

At each node an outward normal to the element is calculated. The outward direction is defined in relation to whether an internal or external problem is being solved. An internal problem is one which the domain under study, Ω, is enclosed by the boundary, Γ. When the domain is outside the boundary, it is an external problem (Figure 5). In the case of potential flow past an elastic structure, the flow would be an external problem and the structure is an internal problem. The ability of the boundary element technique to solve both internal and external solutions with the same boundary geometry allows the user to easily define and track moving boundaries.

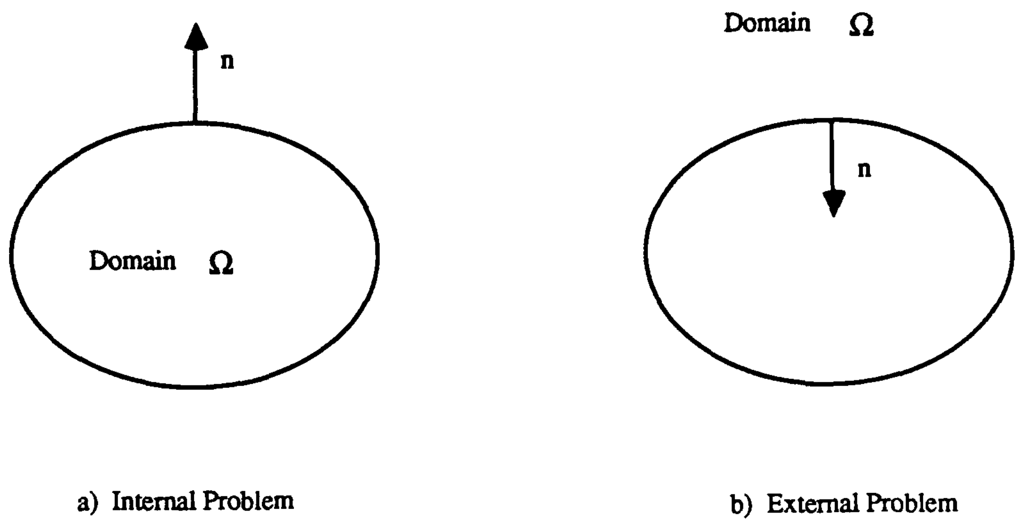


Figure 5. Internal and External Problem Definitions

III. Development of Models

The solution technique first calculates the potential of the fluid about the structure. Next, the velocities and pressures on the surface due to the potential are determined. Finally, the elastic deformations from the pressure field are found. A direct boundary element formulation was used for the potential, velocity and deformation models. This research was limited to a two-dimensional problem, and the following model development will only discuss two-dimensional solutions. A three-dimensional problem is conceptually solved in a similar manner.

Potential Flow

The beginning equation of the direct method boundary element formulation for potential flow is:

$$c_i(\xi)\, u_i(\xi) + \int_{\Gamma} u(x)\, q^{*}(\xi,x)\, d\Gamma(x) = \int_{\Gamma} q(x)\, w^{*}(\xi,x)\, d\Gamma(x) \qquad (15)$$

where;

ξ are the locations of a continuous distribution of sources throughout domain Ω

x are the coordinates for the point where the potential is being evaluated

Γ is the boundary of the domain

i is a point where the fundamental solution is applied, and is called a nodal point

$u(x)$ is the unknown and is the potential at point x

$q = du/dn$

$w^*(\xi,x)$ is the weighted residual weighting function

$q^*(\xi,x) = dw^*/dn$

The fundamental solution to Laplace's equations is used as the weighting function w^*. This solution is the free space Green's function for the governing equation.

For two-dimensional problems the solution is:

$$w^* = \frac{1}{2\pi} \ln\left(\frac{1}{r}\right) \tag{16}$$

Equation 15 has been divided through by 2 π so that the 2 π appears in the denominator of w* and q*. The distance between ξ and x is shown in Figure 6 and is determined by:

$$r(\xi,x) = |\xi - x| = \sqrt{(x(\xi) - x(x))^2 + (y(\xi) - y(x))^2} \tag{17}$$

for a Cartesian two-dimensional system.

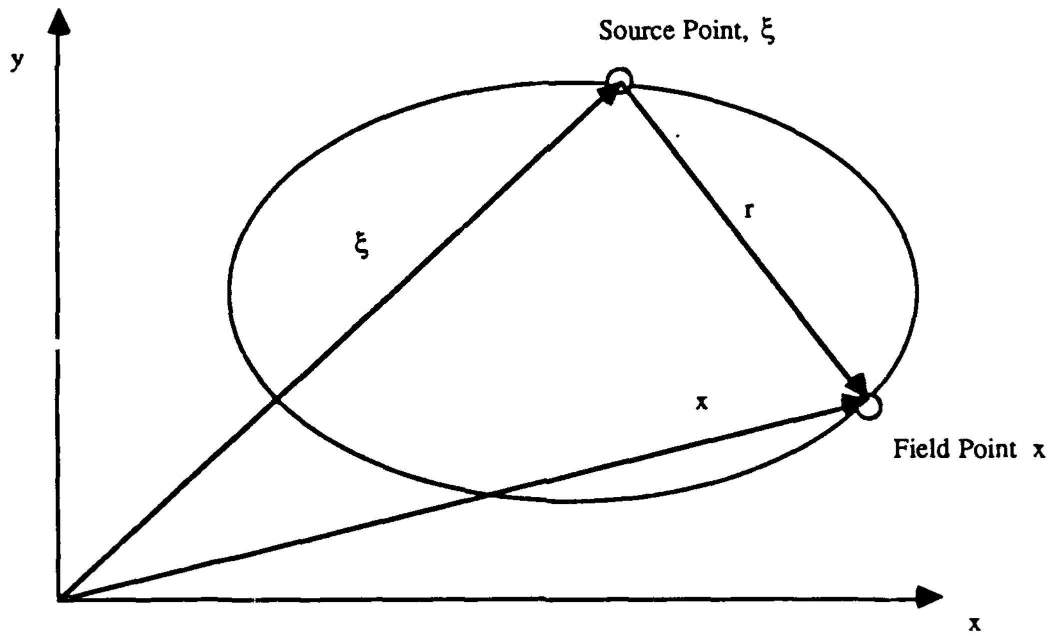


Figure 6. Source and Field Points

When the boundary is discretized into elements Equation 15 becomes:

$$c_i u_i + \sum_{j=1}^{N} \int_{\Gamma_j} u\, q^* \, d\Gamma = \sum_{j=1}^{N} \int_{\Gamma_j} w^* \, q \, d\Gamma \tag{18}$$

where N is the number of geometric elements and Γ_j is the length of element j.

An elemental coordinate system is set up for each element as shown in Figure 7. The values of u and q vary over the element as a function of the coordinate η.

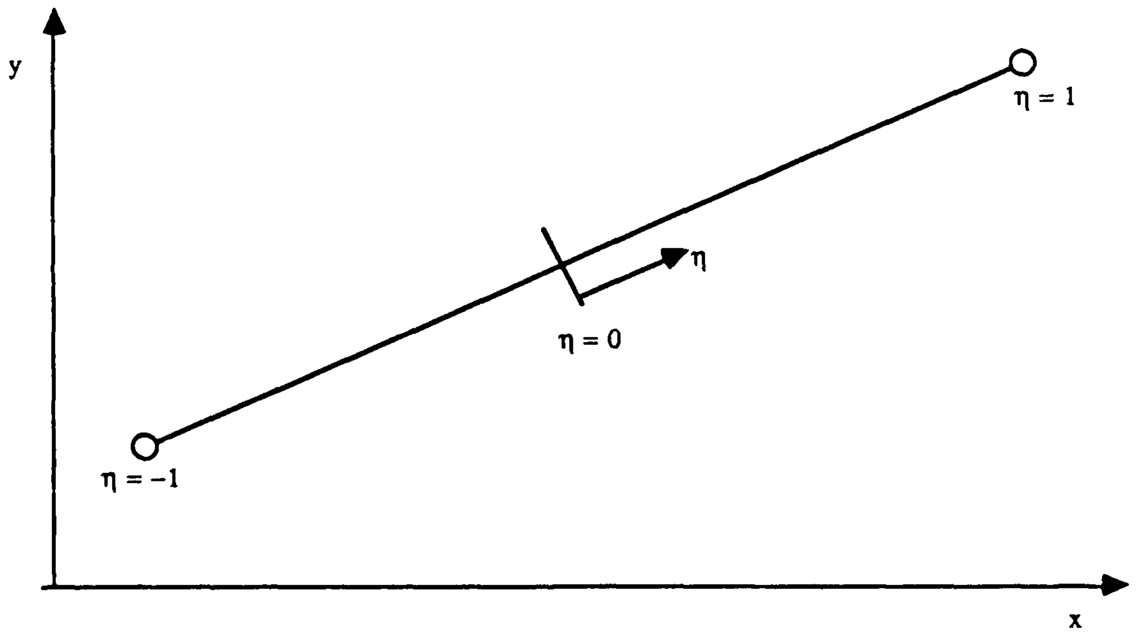


Figure 7. Elemental Coordinate System

The functions u and q may be approximated with shape, or interpolation, functions such that:

$$u(\eta) = N_1(\eta)u_1 + N_2(\eta)u_2 + \ldots + N_r(\eta)u_r = \left[N_1(\eta), N_2(\eta), \;\ldots\; N_r(\eta) \right] \begin{Bmatrix} u_1 \\ u_2 \\ \ldots \\ u_r \end{Bmatrix} \tag{19}$$

$$q(\eta) = N_1(\eta)q_1 + N_2(\eta)q_2 + \ldots + N_r(\eta)q_r = \left[N_1(\eta), N_2(\eta), \ \ldots \ N_r(\eta)\right] \begin{Bmatrix} q_1 \\ q_2 \\ \ldots \\ q_r \end{Bmatrix} \qquad (20)$$

where r is the order of the function modeling (i.e., second, third,etc). Appendix C provides a brief overview of shape functions. Notice that u_r and q_r are the amplitudes of the functions at the node points and are no longer functions of η. Therefore, they may be taken outside the integrals.

The integrals are with respect to the boundary Γ, but u and q are functions of η. A Jacobian is used to transform the integral as detailed in Appendix C ;

$$dG = |G| \, d\eta \qquad (21)$$

By using Equations 19, 20 and 21, Equation 18 now becomes:

$$c_i u_i + \sum_{j=1}^{N} \left(\int_{\Gamma_j} \left[N_1(\eta), N_2(\eta), \ \ldots \ N_r(\eta)\right] q^* \, |G| \, d\eta \right) \begin{Bmatrix} u_1 \\ u_2 \\ \ldots \\ u_r \end{Bmatrix}_j = \qquad (22)$$

$$\sum_{j=1}^{N} \left(\int_{\Gamma_j} \left[N_1(\eta), N_2(\eta), \ \ldots \ N_r(\eta)\right] w^* \, |G| \, d\eta \right) \begin{Bmatrix} q_1 \\ q_2 \\ \ldots \\ q_r \end{Bmatrix}_j$$

Over each element j there are r nodes. Globally, there are $m = j \times r$ total nodes on the boundary (Fig 8). The integral on the left-hand side of Equation 22 can be written as:

$$\int_{\Gamma_j} \left[N_1(\eta), N_2(\eta), \ \ldots \ N_r(\eta)\right] q^* |G| \, d\eta \begin{Bmatrix} u_1 \\ u_2 \\ \ldots \\ u_r \end{Bmatrix}_j = \left[\widehat{h^1_{ij}}, \widehat{h^2_{ij}}, \ldots \widehat{h^r_{ij}} \right] \begin{Bmatrix} u_1 \\ u_2 \\ \ldots \\ u_r \end{Bmatrix}_j \qquad (23)$$

where the $\widehat{h_{ij}^{k}}$ are influence coefficients defining the interaction between the point i under consideration and a particular node k on element j. The same can be done for the right-hand side integral, so that Equation 22 can now be written as:

$$c_i u_i + \sum_{j=1}^{N} \widehat{h_{ij}^{k}} u_k = \sum_{j=1}^{N} g_{ij}^{k} q_k \tag{24}$$

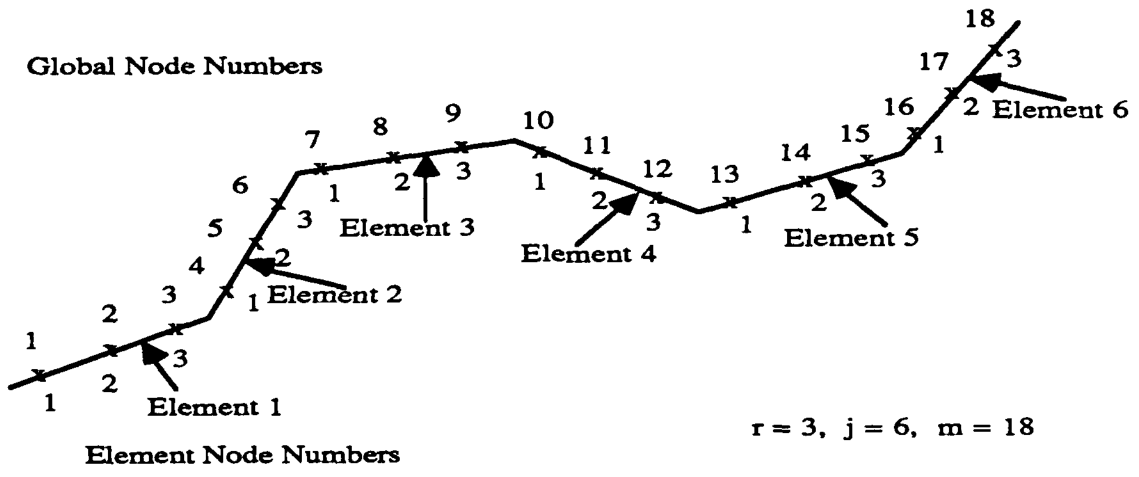


Figure 8. Elemental and Global Node Numbering

When summed over all the elements, equation 24 can be written;

$$c_i u_i + \left[\widehat{H_{i1}}, \widehat{H_{i2}}, \ldots, \widehat{H_{im}}\right] \begin{Bmatrix} u_1 \\ u_2 \\ \cdots \\ u_m \end{Bmatrix} = \left[G_{i1}, G_{i2}, \ldots, G_{im}\right] \begin{Bmatrix} q_1 \\ q_2 \\ \cdots \\ q_m \end{Bmatrix} \tag{25}$$

where m equals the total number of nodes around the boundary in the global numbering system and:

$$\widehat{H_{im}} = \int_{\Gamma} N_{nj}(\eta)\, q^{*}(\xi,x)\, |G|\, d\eta \tag{26}$$

$$G_{im} = \int_{\Gamma} N_{nj}(\eta)\, w^{*}(\xi,x)\, |G|\, d\eta \qquad (27)$$

where the index i is on the node i and nj is the nth node on element j in the global numbering system.

Using Figure 8 for reference, $\widehat{H_{1\,15}}$ is the influence at global node 15 (n = 3, j = 5) due to point 1. The $c_i u_i$ term may be added to the $\widehat{H}$ matrix such that:

$$\begin{aligned} H_{ij} &= \widehat{H_{ij}} && \text{for } i \neq j \\ H_{ij} &= \widehat{H_{ij}} + c_i && \text{for } i = j \end{aligned} \qquad (28)$$

Equation 25 is repeated for each node i (m total) and produces a matrix equation;

$$\mathbf{H}\ \mathbf{u} = \mathbf{G}\ \mathbf{q} \qquad (29)$$

For a smooth boundary, it can be shown that the coefficient, c_i, is equal to 1/2 as discussed in Chapter 2. If the surface is not smooth, it will not be equal to 1/2. The diagonal term of **H** may also be calculated by using the fact that when a uniform potential is applied on the whole body the sum of the values of q's must be zero. For internal or closed problems ;

$$\mathbf{H}\,\mathbf{I} = \mathbf{0} \qquad (30)$$

so that the sum of each row in **H** should be zero (3:111). Therefore,

$$H_{ii} = -\sum_{j=1}^{N} H_{ij} \qquad i = 1, 2, \ldots\ldots, N,\ i \neq j \qquad (31)$$

For exterior, or unbounded, problems it can be shown (3:112) that:

$$H_{ii} = -\sum_{j=1}^{N} H_{ij} + 1 \qquad i = 1, 2, \ldots\ldots, N,\ i \neq j \qquad (32)$$

The boundary conditions, q, can be calculated using the assumption of zero flow through the boundary. This means that the source must produce equal and opposite flow to the flow at the boundary, or:

$$q_i = -\overline{n_i} \cdot V_{\infty} \qquad (33)$$

where n_i is the outward unit normal at point i and V_∞ is the flow condition at infinity.

The matrices H and G may be calculated numerically using numerical quadrature for the integral terms. Equation 29 is a m x m system of equations. As the integration progresses around the boundary, for each node i there will be a panel for which the integration is singular (r approaches and equals 0). Care must be taken to properly integrate these singular panels. Note that ;

$$\overset{\bullet}{q}(\xi,x) = \frac{\partial \overset{\bullet}{w}}{\partial n} = \frac{\partial}{\partial n}\left(\frac{1}{2\pi}\ln\frac{1}{r}\right) = \frac{1}{2\pi}\left(n \cdot \nabla\left(\ln\frac{1}{r}\right)\right) \tag{34}$$

On a flat singular panel the normal is perpendicular to the panel and therefore;

$$n \cdot \nabla\left(\ln\frac{1}{r}\right) = 0 \tag{35}$$

Therefore, the $\widehat{H}$ terms are zero on the singular panel. With higher order geometric modeling this is not the case and this term must be addressed.

Velocity and Pressure Distribution

Once Equation 29 is solved for the potential at the nodes, the velocity and pressure distribution at the nodes can be determined. The components of the perturbation velocity at node i are calculated using:

$$\frac{\partial u(\xi)}{\partial x_i(x)} = \frac{1}{c_i}\left\{\int_\Gamma q(x)\frac{\partial \overset{\bullet}{w}(\xi,x)}{\partial x_i(x)}\,d\Gamma(x) - \int_\Gamma u(x)\frac{\partial \overset{\bullet}{q}(\xi,x)}{\partial x_i(x)}\,d\Gamma(x)\right\} \tag{36}$$

A m x m system of equations may be set up for Equation 36 similarly to that described above, and numerical quadrature used to calculate the terms of the matrices. The functions q(x) and u(x) are known from previous calculations so that the velocity may be solved. The components of the perturbation velocity in the x and y directions are:

$$u = \frac{\partial u(\xi)}{\partial x(x)} \quad \text{and} \quad v = \frac{\partial u(\xi)}{\partial y(x)} \tag{37}$$

and the total perturbation velocity is calculated by:

$$V_p = \sqrt{u^2 + v^2} \tag{38}$$

The total velocity at node i is equal to:

$$V = V_\infty - V_p \tag{39}$$

The pressure field about the body is computed using:

$$C_p = 1 - \left(\frac{V}{V_\infty}\right)^2 \tag{40}$$

$$P = \frac{1}{2}\rho_\infty V_\infty^2 C_p \tag{41}$$

$$P_x = P \cdot n_x \quad \text{and} \quad P_y = P \cdot n_y \tag{42}$$

P_x and P_y are the components of the pressure acting on the surface or the surface tractions.

Elastic Structure

The development of the boundary element formulation for the elastic structure is quite similar to that of the potential flow. The structure consists of subregions formed by an outer boundary, inner boundary and connecting elements as shown in Figure 9. The connecting elements allow the effect of a force at one point on the boundary to be transferred around the structure. For this investigation, the same number of nodes were used on the outside, inside and connecting elements although this is not required.

The beginning statement for each subregion is:

$$c_{ij}u_j + \int_\Gamma p^*_{ij}u_j\,d\Gamma = \int_\Gamma u^*_{ij}P_j\,d\Gamma + \int_\Omega u^*_{ij}b_j\,d\Omega \tag{43}$$

where

u_j is now the unknown displacements in the j direction

P_j is the traction or natural boundary conditions in the j direction (the pressures calculated above)

b_j is the body force in the j direction and is assumed zero for this research.

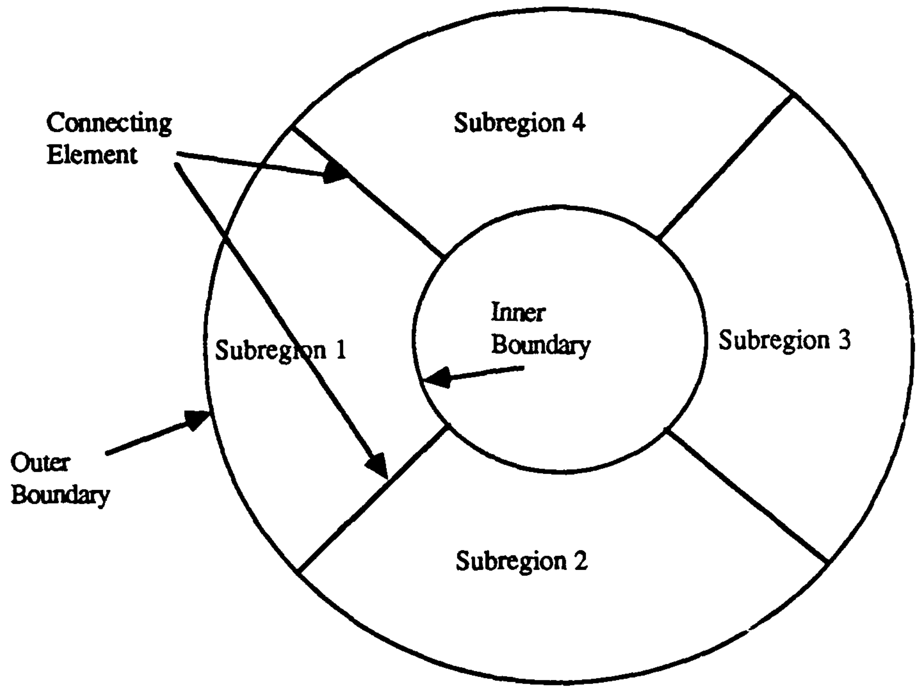


Figure 9. Structure Geometry

The fundamental solution w* for a two-dimensional plane strain problem is (2:187):

$$w^*_{ij} = \frac{1}{8\pi(1-\nu)G}\left\{(3-4\nu)\ln(r)\,\delta_{ij} + \frac{\partial r}{\partial x_i}\frac{\partial r}{\partial x_j}\right\} \tag{44}$$

The solution p* is defined as:

$$p^*_{ij} = \frac{-1}{4\pi(1-\nu)r}\left\{\left[(1-2\nu)\,\delta_{ij} + 2\frac{\partial r}{\partial x_i}\frac{\partial r}{\partial x_j}\right]\frac{\partial r}{\partial n} - (1-2\nu)\left(\frac{\partial r}{\partial x_i}n_j - \frac{\partial r}{\partial x_j}n_i\right)\right\} \tag{45}$$

As with the potential flow equations, the functions u and q are represented with shape functions and the integrals transformed from the dΓ to the dη system.

Equation 43 can be written:

$$c_i u_i + \left[\hat{h}_{i1}, \hat{h}_{i2}, \ldots, \hat{h}_{ii}, \ldots, \hat{h}_{im}\right] \begin{Bmatrix} u_1 \\ u_2 \\ \cdots \\ u_i \\ \cdots \\ u_m \end{Bmatrix} = \left[g_{i1}, g_{i2}, \ldots, g_{ii}, \ldots, g_{im}\right] \begin{Bmatrix} p_1 \\ p_2 \\ \cdots \\ p_i \\ \cdots \\ p_m \end{Bmatrix} \qquad (46)$$

m is the total number of nodes around the outer boundary and is equal to the nodes around the inner boundary and also around the connecting elements. The approximating functions of u and q are modeled with the same order, r, on all elements. Therefore, m = 3 x r nodes around the region. Note that the matrix terms are 2 x 2 for the two-dimensional case because they contain the terms;

$$\hat{h}_{ij} = \begin{bmatrix} h_{ij\,xx} & h_{ij\,xy} \\ h_{ij\,yx} & h_{ij\,yy} \end{bmatrix} \qquad g_{ij} = \begin{bmatrix} g_{ij\,xx} & g_{ij\,xy} \\ g_{ij\,yx} & g_{ij\,yy} \end{bmatrix} \qquad (47)$$

For the three-dimensional case the matrices would be 3 x 3.

Equation 46 produces a 1x(3xrx2) system of equations for each node i. Equation 46 is calculated twice for each of the nodes on the connecting element; once for the element to the left and once for the element to the right. In this way the matrices have (4xrx2) rows. The final system of equastions is a (4xrx2) x (3xrx2) system.

The coefficient c_i may be handled similarly to Equation 29 to form the system of equations:

$$\mathbf{H\,u = G\,P} \qquad (48)$$

The c_i coefficient may be calculated using rigid-body considerations in a bounded body (3:201). If a unit rigid-body displacement is made in any direction Equation 48 becomes:

$$\mathbf{H\,I}_d = 0 \qquad (49)$$

where $\mathbf{I}_d$ is a vector of unit displacements in the d direction. The diagonal terms of **H** become:

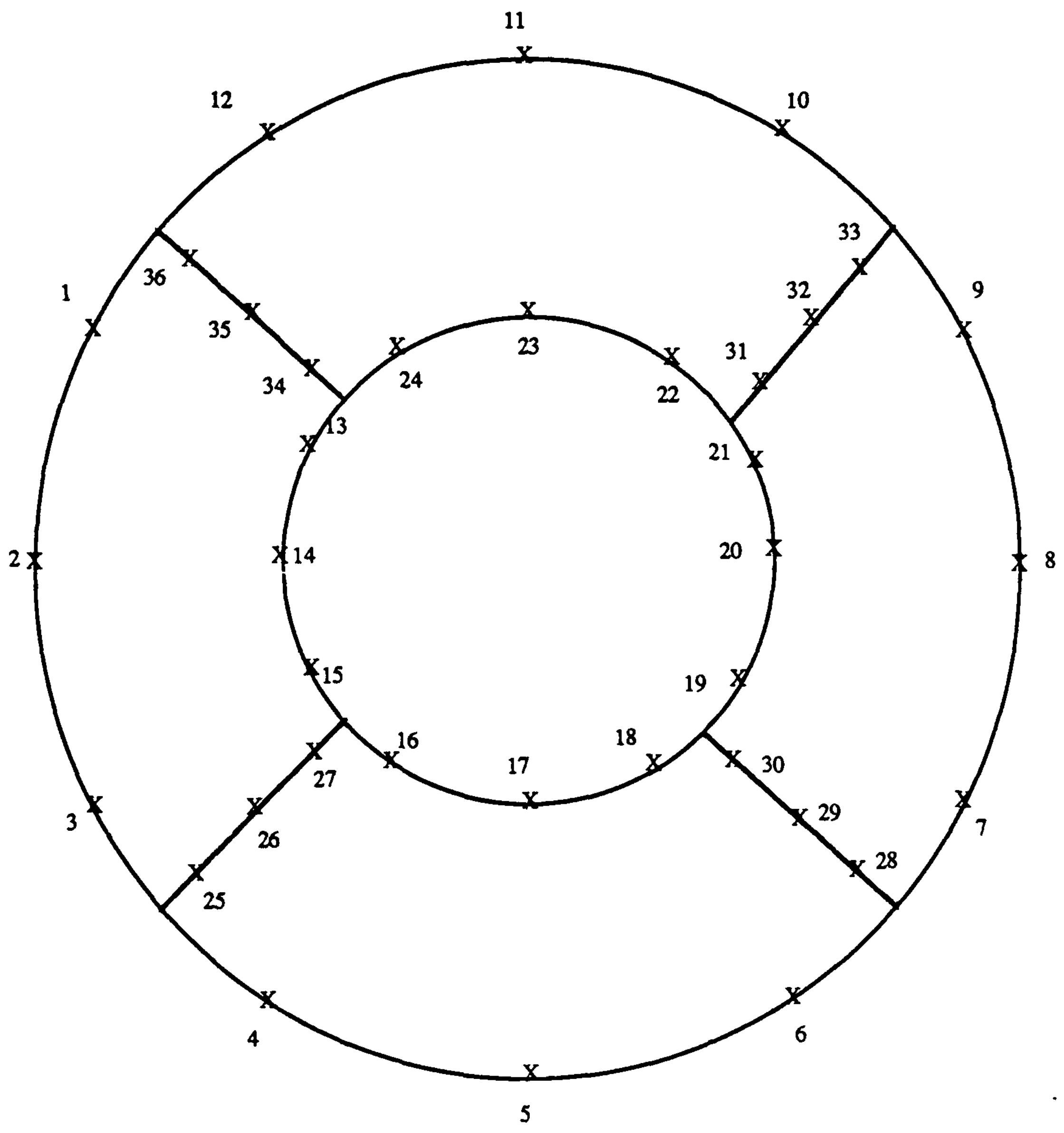
$$\mathbf{h}_{ii} = -\sum_{i=1}^{N} \mathbf{h}_{ij} \qquad \text{for } i \neq j \tag{50}$$

<u>Assembly of Subregions.</u> For each subregion in the structure the system in Equation 48 may be developed. To solve for the deformations about the entire body, the systems of equations must be assembled into a global system of equations. This is done by renumbering the nodes around the structure. The nodes around the outside are numbered 1 to m, the nodes about the inside m+1 to 2m, and the nodes on the connecting elements 2m+1 to 3m (Figure 10). The **H** and **G** matrices are now (8xm) x (6xm). The global system of equations is:

$$\begin{bmatrix}
H^{out}_{out\,xx} & H^{out}_{in\,xx} & H^{out}_{con\,xx} & H^{out}_{out\,xy} & H^{out}_{in\,xy} & H^{out}_{con\,xy} \\
H^{in}_{out\,xx} & H^{in}_{in\,xx} & H^{in}_{con\,xx} & H^{in}_{out\,xy} & H^{in}_{in\,xy} & H^{in}_{con\,xy} \\
H^{con}_{out\,xx} & H^{con}_{in\,xx} & H^{con}_{con\,xx} & H^{con}_{out\,xy} & H^{con}_{in\,xy} & H^{con}_{con\,xy} \\
H^{con}_{out\,xx} & H^{con}_{in\,xx} & H^{con}_{con\,xx} & H^{con}_{out\,xy} & H^{con}_{in\,xy} & H^{con}_{con\,xy} \\
H^{out}_{out\,yx} & H^{out}_{in\,yx} & H^{out}_{con\,yx} & H^{out}_{out\,yy} & H^{out}_{in\,yy} & H^{out}_{con\,yy} \\
H^{in}_{out\,yx} & H^{in}_{in\,yx} & H^{in}_{con\,yx} & H^{in}_{out\,yy} & H^{in}_{in\,yy} & H^{in}_{con\,yy} \\
H^{con}_{out\,yx} & H^{con}_{in\,yx} & H^{con}_{con\,yx} & H^{con}_{out\,yy} & H^{con}_{in\,yy} & H^{con}_{con\,yy} \\
H^{con}_{out\,yx} & H^{con}_{in\,yx} & H^{con}_{con\,yx} & H^{con}_{out\,yy} & H^{con}_{in\,yy} & H^{con}_{con\,yy}
\end{bmatrix}
\begin{Bmatrix} u_{out\,x} \\ u_{in\,x} \\ u_{con\,x} \\ u_{out\,y} \\ u_{in\,y} \\ u_{con\,y} \end{Bmatrix} = \tag{51}$$

$$\begin{bmatrix}
G^{out}_{out\,xx} & G^{out}_{in\,xx} & G^{out}_{con\,xx} & G^{out}_{out\,xy} & G^{out}_{in\,xy} & G^{out}_{con\,xy} \\
G^{in}_{out\,xx} & G^{in}_{in\,xx} & G^{in}_{con\,xx} & G^{in}_{out\,xy} & G^{in}_{in\,xy} & G^{in}_{con\,xy} \\
G^{con}_{out\,xx} & G^{con}_{in\,xx} & G^{con}_{con\,xx} & G^{con}_{out\,xy} & G^{con}_{in\,xy} & G^{con}_{con\,xy} \\
G^{con}_{out\,xx} & G^{con}_{in\,xx} & G^{con}_{con\,xx} & G^{con}_{out\,xy} & G^{con}_{in\,xy} & G^{con}_{con\,xy} \\
G^{out}_{out\,yx} & G^{out}_{in\,yx} & G^{out}_{con\,yx} & G^{out}_{out\,yy} & G^{out}_{in\,yy} & G^{out}_{con\,yy} \\
G^{in}_{out\,yx} & G^{in}_{in\,yx} & G^{in}_{con\,yx} & G^{in}_{out\,yy} & G^{in}_{in\,yy} & G^{in}_{con\,yy} \\
G^{con}_{out\,yx} & G^{con}_{in\,yx} & G^{con}_{con\,yx} & G^{con}_{out\,yy} & G^{con}_{in\,yy} & G^{con}_{con\,yy} \\
G^{con}_{out\,yx} & G^{con}_{in\,yx} & G^{con}_{con\,yx} & G^{con}_{out\,yy} & G^{con}_{in\,yy} & G^{con}_{con\,yy}
\end{bmatrix}
\begin{Bmatrix} q_{out\,x} \\ q_{in\,x} \\ q_{con\,x} \\ q_{out\,y} \\ q_{in\,y} \\ q_{con\,y} \end{Bmatrix}$$

where $\mathbf{H}^i{}_{jlk}$ and $\mathbf{G}^i{}_{jlk}$ are the influence coefficients at node i due to panel j in the lk direction.The boundary forces around the boundaries are the pressures from the fluid and have been calculated previously. The boundary tractions along the connecting surfaces are unknowns and must be calculated with the unknown deformations. The columns of **G** and the corresponding rows of **P** are transferred to the left-hand side of the equations forming a (8xm) x (8xm) matrix with (8xm) unknowns. This final set of equations may now be solved for the unknown deformations and internal tractions.



r = 3, j = 4, m = 12, 3 x m = 36

Figure 10. Global Node Numbering for Structure

IV. Implementation

Once the potential, velocity and elastic deformation models were developed, they were coded into a Fortran computer program. Several simplifications, assumptions and numerical approximations were included in the program code.

Geometry

The structure used during this research was a two-dimensional thick-walled cylinder with an outer and inner radius (Figure 11). The geometry was modeled as flat elements with geometry nodes at the left and right ends of the elements ($\eta=-1$, $\eta=1$). The number of elements around the cylinder could be varied so that the cylinder could be modeled very crudely with 8 elements as shown in Figure 11, or finely with up to 150 elements. The number of outer boundary elements, inner boundary elements and connecting elements were equal for ease of geometry definition. The elements were numbered in a counterclockwise direction with the structure domain remaining on the left-side as one travels around the outer boundary, and on the right-side along the inner boundary. The connecting elements were specified so that the left end would be on the outer boundary and the right end on the inner boundary.

The element Jacobian was calculated as detailed in Appendix C, and the outward normals calculated by:

$$n_x = \frac{-\partial y/\partial \eta}{\text{Jacobian}} \quad \text{and} \quad n_y = \frac{\partial x/\partial \eta}{\text{Jacobian}} \tag{52}$$

Because of the counterclockwise setup of the elements, the outward normal on the outer boundary is into the structure and therefore the correct sign for the potential flow calculations. For the elastic deformation calculations a negative sign must be applied to the normal so that it may be an outward normal for the structure. The inner boundary normal is in the correct direction for the elastic deformations. The correct direction of the connecting elements outward normals were determined in the same manner.

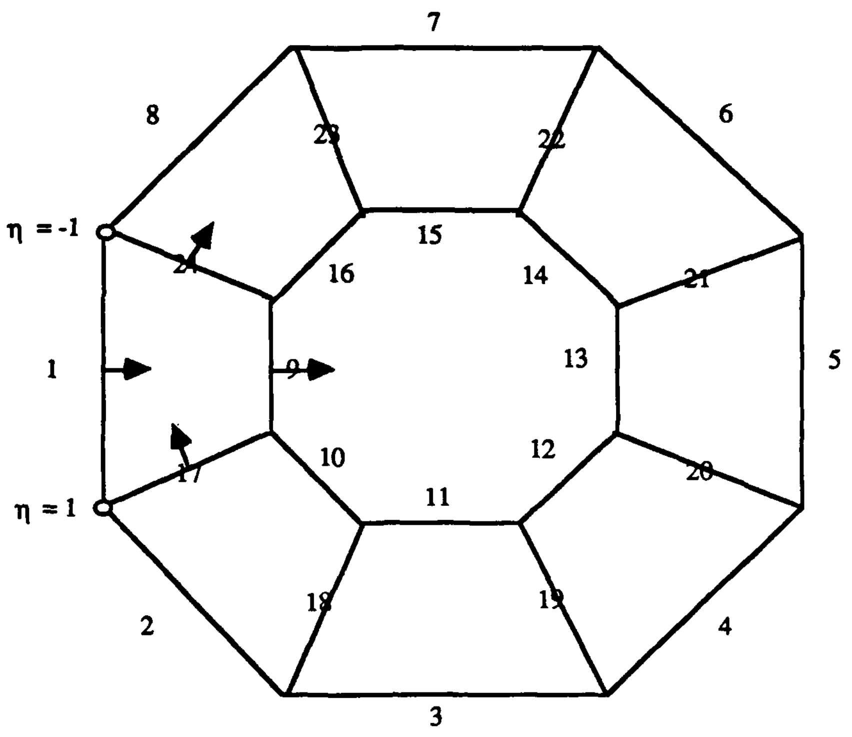


Figure 11. Geometry Modeled With 8 Flat Elements

The order of the potential, pressure and deformation approximation was equal and could be specified from constant (1 node) to 4th order (5 nodes). Therefore, the potential, pressure and deformation were all calculated at the same node. The general BEM formulation does not require this assumption. The distance of the end nodes to the element corner, δ, was specified by the user. In this way the effect of the distance between the end node and the corner could be investigated. The end node was never placed directly on the corner (δ = 0). As previously discussed there are several difficulties that arise if the node is placed directly on the corner which is a geometry slope discontinuity. The remaining nodes were spaced equally between the end nodes (Figure 12) with the same number of nodes used on the outer, inner and connecting boundary elements.

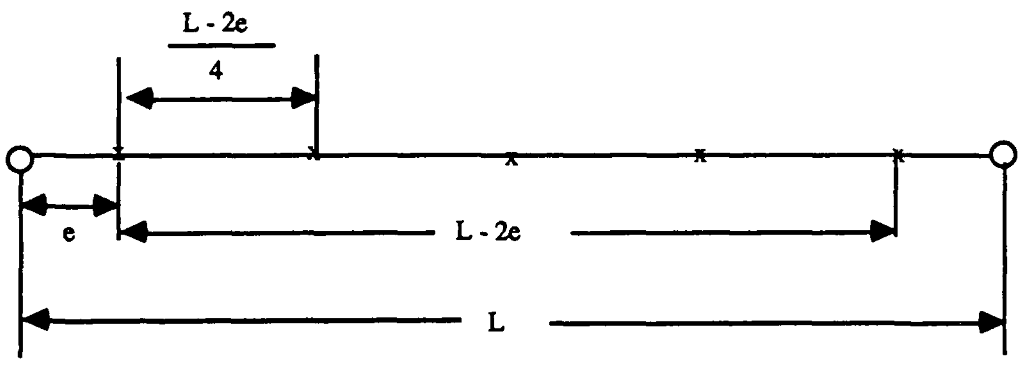


Figure 12. Nodal Spacing Within An Element

Once the deformations at the node points were calculated, the deformations at the ends of each panel were determined using the shape functions. For example, the deformations at the left end of a panel are:

$$\begin{aligned} u_x(\eta = -1) &= N_1(-1)\, u_{x1} + N_2(-1)\, u_{x2} + \ldots + N_r(-1)\, u_{xr} \\ u_y(\eta = -1) &= N_1(-1)\, u_{y1} + N_2(-1)\, u_{y2} + \ldots + N_r(-1)\, u_{yr} \end{aligned} \qquad (53)$$

where $N_i(-1)$ is the shape function at η=-1 for element node i, and u_{xi} and u_{yi} are the deformations at node i on the element. The deformations at the right-end of the element were calculated in a similar manner. Because the discontinuous slope at the corners causes stress concentrationsat the corners the deformations are not continuous from element to element. In order to force continuity of the new geometry, the deformations as each corner were averaged with the deformation of the right panel and the deformation of the left panel. The averaged deformations were applied to the original geometry resulting in the new geometry boundaries. Once the new outer and inner geometry was calculated, the connecting element geometry was recalculated.

Potential Flow

Potential flow acts only on the outer boundary. There is no flow inside the cylinder. The potential flow subroutine consists of an outer loop on the number of nodes and an inner loop on the number of elements. For each node the routine sets up Equation 15 and numerically integrates over all the elements.

The functions w* and q* are defined as:

$$w^* = \left(\frac{1}{2\pi}\right) \ln\left(\frac{1}{r}\right) \qquad \text{and} \qquad q^* = \frac{\partial w^*}{\partial n} = \left(\frac{1}{2\pi}\right)\left(\frac{n_i r_i - n_j r_j}{r^2}\right) \tag{54}$$

On the non-singular elements the integration was performed with a Gaussian quadrature:

$$\int_{-1}^{1} f(\eta)\, d\eta = \sum_{i=1}^{N} f(\eta_i)\, w_i \tag{55}$$

where N is the order of the quadrature, η_i are the quadrature values and w_i the quadrature weights. The order of the quadrature used was determined by comparing the numerical integration results with analytical results as the element end points were moved towards the corner. In this way it was found that a 12th order quadrature was required to obtain accurate integrations for small values of δ. This was also verified by comparison of the diagonal terms of the **H** matrix as the quadrature order was increased. Because the geometry was always smooth at the nodes, the diagonal term should be .5. A 12th order quadrature was required to obtain .5 as the end nodes came very close to the corners (δ = .01). This high order is due to the 1/r term which causes r to become very small as the node is moved to the end of the panel.

The singular element integrations were performed with a different quadrature formula. The singular terms of the **H** matrix (q* integral) vanish because the normal is perpendicular to the panel for flat panels (Eq 34, 35). For the **G** matrix, a one-dimensional logarithmic Gaussian quadrature formula is used (3:449). Because the formula is designed for an integral from 0 to 1 with the singularity at 0, the integral is broken into two separate integrals:

$$\int_{-1}^{1} [N(\eta)]\left(\frac{1}{2\pi}\right) \ln\left(\frac{1}{r}\right)|G|\, d\eta = \int_{-1}^{r=0} f(\eta) \ln\left(\frac{1}{r}\right) d\eta + \int_{r=0}^{1} f(\eta) \ln\left(\frac{1}{r}\right) d\eta \tag{56}$$

$$= \int_{0}^{1} -f(\bar{\eta}) \ln\left(\frac{1}{r}\right)(1 + \eta(r=0))\, d\bar{\eta} + \int_{0}^{1} f(\bar{\bar{\eta}}) \ln\left(\frac{1}{r}\right)(1 - \eta(r=0))\, d\bar{\bar{\eta}}$$

where

$$\bar{\eta} = \frac{\eta - \eta(r=0)}{-1 - \eta(r=0)} \quad \text{and} \quad \bar{\bar{\eta}} = \frac{\eta - \eta(r=0)}{1 - \eta(r=0)} \tag{57}$$

S is now defined as the location of the singularity on the panel ($\eta(r{=}0)$). On the singular element

$$r = |S - \eta|\frac{\text{Length}}{2} \tag{58}$$

where Length is the length of the panel. Substituting Equation 58 into Equation 56 results in:

$$\int_{-1}^{1} f(\eta)\ln\left(\frac{1}{r}\right)d\eta = \tag{59}$$

$$\int_{0}^{1} f(\bar{\eta})\ln\left(\frac{1}{r}\right)(1+S)\,d\bar{\eta} - \int_{-1}^{1} f(\bar{\bar{\bar{\eta}}})\ln\left(|1+S|\frac{\text{Length}}{2}\right)(1+S)\,.5\,d\bar{\bar{\bar{\eta}}}$$

$$+ \int_{0}^{1} f(\bar{\bar{\eta}})\ln\left(\frac{1}{\bar{\bar{\bar{\eta}}}}\right)(1-S)\,d\bar{\bar{\eta}} - \int_{-1}^{1} f(\bar{\bar{\bar{\eta}}})\ln\left(|S-1|\frac{\text{Length}}{2}\right)(1-S)\,.5\,d\bar{\bar{\bar{\eta}}}$$

where

$$\bar{\bar{\bar{\eta}}} = 2\bar{\eta} - 1 \quad \text{and} \quad \eta = 2\bar{\bar{\eta}} - 1 \tag{60}$$

Using Equation 59 the singular terms of the **G** matrix were calculated. Once the **H** and **G** matrices are filled, the program calculates the diagonal terms of the **H** matrix using Equation 32, and the boundary condition matrix **q** using no flow through the boundary (Equation 33).

Velocity and Pressure Field

The velocity and pressure subroutine is similar to the potential flow routine in that it performs an outer loop on the number of nodes and an inner loop on the number of elements. The fundamental solutions are now:

$$\frac{\partial w^*}{\partial x(x)} = \frac{-1}{2\pi}\frac{r_i}{r^2} \quad \text{and} \quad \frac{\partial w^*}{\partial y(x)} = \frac{-1}{2\pi}\frac{r_j}{r^2} \tag{61}$$

and

$$\frac{\partial q^*}{\partial x(x)} = \frac{1}{2\pi}\frac{r^2 n_i - (r_i n_i - r_j n_j)\, 2\, r_i}{r^4} \quad \text{and} \quad \frac{\partial q^*}{\partial y(x)} = \frac{1}{2\pi}\frac{r^2 n_j - (r_i n_i - r_j n_j)\, 2\, r_j}{r^4} \tag{62}$$

where the r_i and r_j terms are the components of r in the x and y direction. The non-singular integrals are approximated by a 12th order Gaussian quadrature.

On the singular panels the following approximations are made (Figure 13):

$$r = \left| x - \eta \right| \frac{\text{Length}}{2} \tag{63}$$

$$r_i = \left(\frac{x - \eta}{\left| x - \eta \right|} \right) r \cos\theta \quad \text{and} \quad r_j = \left(\frac{x - \eta}{\left| x - \eta \right|} \right) r \sin\theta \tag{64}$$

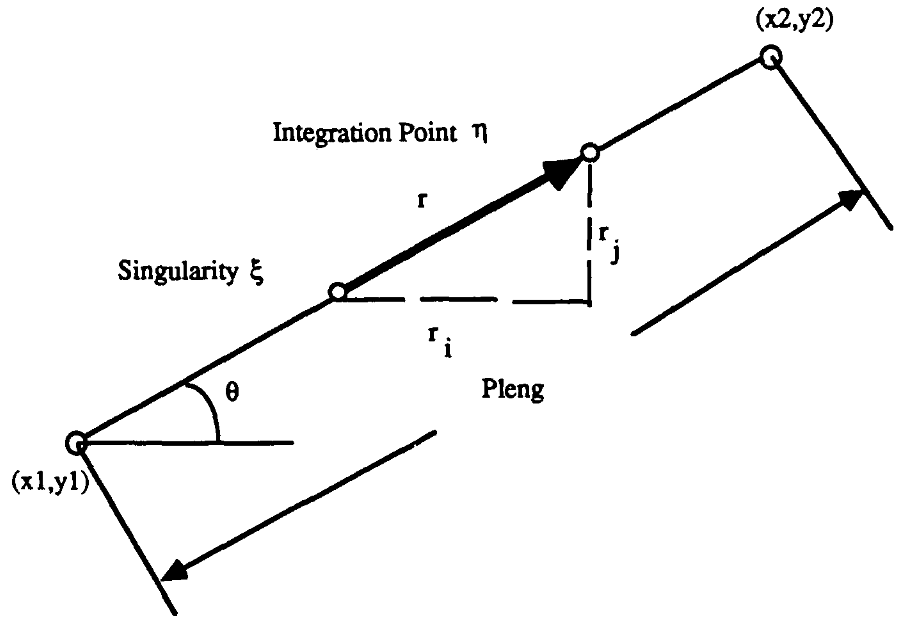


Figure 13. Singular Panel Approximation Definitions

Because r is always postive, a correction factor was required to place the correct sign on the r components. The angle formed by the x-axis and the element is θ and is calculated by:

$$\cos\theta = \frac{x_2 - x_1}{\text{Length}} \quad \text{and} \quad \sin\theta = \frac{y_2 - y_1}{\text{Length}} \tag{65}$$

where (x_2,y_2) are the coordinates of the right end of the panel, and (x_1,y_1) are the coordinates for the left end.

By using Equations 63, 64 and 65 Equation 61 reduces to a 1/r singularity term. A quadrature using a numerical evaluation of Cauchy principal values (3:451) was used for the integrations.

$$\int_{-1}^{1} \frac{f(\eta)}{(\eta - S)} d\eta = \sum_{i=1}^{K} f\left[(-1 - S)\,\eta_i + S\right] w_i + f(S) \ln|-1 - S| \tag{66}$$

$$- \sum_{i=1}^{K} f\left[(1 - S)\,\eta_i + S\right] w_i - f(S) \ln|1 - S|$$

Equation 62 reduces to a $1/r^2$ singularity. From Reference 9 this can be approximated by a quadrature rule using;

$$\int_{-1}^{1} \frac{f(\eta)}{\eta^2} d\eta = \sum_{i=1}^{K} w_i \frac{f(\eta_i)}{\eta_i^2} - \left(\sum_{i=1}^{K} \frac{w_i}{\eta_i^2} + 2\right) f(\eta{=}0) \tag{67}$$

Note that Equation 67 assumes that the singularity is in the center of the element. To use this the singular element was divided into one segment center around the sigularity, and a second segment which does not contain the singularity.

$$\int_{-1}^{1} \frac{f(\eta)}{\eta^2} d\eta = \int_{\text{start}}^{\text{end}} \frac{f(\hat{\eta})}{\hat{\eta}^2} .5\,(\text{end} - \text{start})\, d\hat{\eta} + \text{non-singular integral} \tag{68}$$

where start = start of singular segment and end = end of singular segment. If the singular segment is from -1 to end, then the non-singular integral will be:

$$\text{non-singular integral} = \int_{-1}^{1} \frac{f(\tilde{\eta})}{\tilde{\eta}^2} .5\,(1-\text{end})\,d\tilde{\eta} \quad \text{where} \quad \tilde{\eta} = 2\left(\frac{\eta - \text{end}}{1 - \text{end}}\right) - 1 \qquad (69)$$

If the singular segment is from start to 1, then the non-singular integral will be:

$$\text{non-singular integral} = \int_{-1}^{1} \frac{f(\tilde{\tilde{\eta}})}{\tilde{\tilde{\eta}}^2} .5\,(\text{start}+1)\,d\tilde{\tilde{\eta}} \quad \text{where} \quad \tilde{\tilde{\eta}} = 2\left(\frac{\eta + 1}{\text{start} + 1}\right) - 1 \qquad (70)$$

The second integral may be solved using a normal Gaussian quadrature while the singular segment may use Equation 67.

Both the boundary conditions and the potential are known at the nodal points. Equation 36 is solved at each node in the x direction for the x-component of perturbation velocity and in the y direction for the y-component of perturbation velocity. The total perturbation velocity is subtracted from the oncoming flow velocity at each node to determine the final velocity at each node. The pressure field on the outer surface is then calculated using Equations 40, 41 and 42. By changing the density of the fluid, the user may increase or decrease the magnitude of the pressure on the cylinder.

For the pressure field on the inner boundary the inner pressure was assumed constant, and was set equal to a pressure on an outer node. This approximates the cylinder being vented to the outside. The node could be chosen by the user, and was selected as a node on the back of the cylinder where the pressure approached stagnation pressure. Once a vented node was specified, the pressure field on the inner boundary was determined.

Elastic Deformations

The elastic deformations were determined in a nested three step process. The outer step is on the subregions of the structure, the next step on the nodes around the subregion, and the innermost step on the elements around the subregion. The fundamental equations, w^* and q^*, are given in Equations 44 and 45. For the non-singular integrals the 12th order quadrature is used. For the singular integrals the equations are reduced to $\ln(1/r)$ and $(1/r)$ singularities whose quadrature forms have been previously discussed. As the routine progresses around a subregion it places the terms of the **H** and **G** matrices into the global numbering system rows and columns. When the routine has finished looping around all the subregions, the matrices are entirely filled. This routine did not attempt to optimize the **H** and **G** matrices. If optimization were to be performed, the matrices would be banded which would reduce the time required for solution. The diagonal terms of the **H** matrix are calculated using Equation 50 and the unknown tractions are moved to the left-hand side of the equation. The system of equations is then solved for the deformations at the nodal points.

Solution of Linear Equations

The matrix system **HU** = **Gq** was solved using a routine found in Reference 2 (2:67). The proper execution of this routine was verified using IMSL routines. IMSL was not used because it requires additional workspace. Because of the large systems of matrix equations used in the program, memory space was a driving factor to the size of the problem solved.

Continuity Considerations

Because the end nodes of the potential and deformation functions were not on the end of the element, the approximating functions were not constrained to be continuous from element to element. To force the potential to be continuous between elements a continuity restriction could be enforced at the corners. At each corner the function u computed using shape functions from the left element equalled the function computed using shape functions from the right element:

$$N_1(-1)u_{1\,rt}+N_2(-1)u_{2\,rt}+\ldots+N_r(-1)u_{r\,rt}=N_1(1)u_{1\,lf}+N_2(1)u_{2\,lf}+\ldots+N_r(1)u_{r\,lf} \tag{71}$$

or:

$$N_1(-1)u_{1\,rt}+N_2(-1)u_{2\,rt}+\ldots+N_r(-1)u_{r\,rt}-N_1(1)u_{1\,lf}-N_2(1)u_{2\,lf}-\ldots-N_r(1)u_{r\,lf}=0 \tag{72}$$

A row of the **Hu** = **Gq** system was replaced on each element by Equation 72. When solved, the new system of equations produced continuous values of the function about the surface. In practice, the order of the approximating function was increased to allow the same number of collocation points on the elements as were used without the continuity restraint. An alternative approach not used here would have been to use a least squares solution by applying a single value decomposition solver. This approach would not have ensured continuity, however, and was rejected in favor of the first approach.

Slope continuity would be desirable. However, since the geometry was modeled as straight line segments the solution was expected to display singular behavior in the vicinity of the corners (8). This is analogous to the stress concentrations expected at sharp corners in elastic solutions. If the geometry model is forced to have continuous slope in the future, it would be desirable to enforce not only continuity of the unknown but also continuity of its slope at the junction of the elements.

The notion of a least squares approach could be applied to higher order elements. This could replace the existing linear terms satisfying the original geometric nodal data and the slope continuity requirement. The new geometric nodal points could then be used to describe the boundary.

Iterative Process

The potential, velocity and pressure field due to the original geometry are calculated the first pass through the routine. The deformations caused by the pressure field (surface tractions) are determined, and the average corner deformations are applied to the original geometry resulting in a deformed geometry.

A new potential, velocity and pressure field are calculated. Beginning with the second iteration the elastic structure is in a pre-stressed condition. If the total pressure field is used to calculate the deformations, it would be equivalent to releasing the stresses in the structure after each iteration and beginning from an initial condition. To model the pre-stressed condition the new surface tractions are subtracted from the old tractions. This difference determines the tractions caused only by the deformations from the previous iteration. The difference is then used to calculate the new deformations and new geometry.

The program iterates in the above manner, calculating new tractions and deformations each iteration. If the structure has adequate stiffness, the incremental deformations will approach zero, and a steady-state condition will be obtained. If the stiffness is not adequate for the flow condition, the iterative process will not reach a steady-state condition.

V. Discussion and Results

The results of the research are presented as solutions of potential, velocity, coefficient of pressure and deformation. Because the solution of potential flow about a cylinder is well known, the results of the BEM potential flow are compared to theory to validate the potential flow code. Once the potential solutions have been discussed, the results of the elastic deformations are discussed and are compared to results from a FEM solution. Finally, the iteration of the fluid-structure problem is discussed.

Effect of Element Size on Solution

The structure could be modeled with 8 to 120 elements, and as the number of elements increased the size of each element decreased. As the element size decreased, the geometry approached that of a cylinder. Therefore, it would be expected that as the size of elements decreased the BEM solution for the velocity and pressure would approach theoretical values about a cylinder.

The coefficient of pressure was the parameter chosen to discuss because it is the end result of the potential and velocity calculations. Figures A1, A2 and A3 illustrate the effect of element size on the solution of coefficient of pressure (Cp) for constant, linear and parabolic modeling of potential and velocity. The solution is presented for 8 elements, 40 elements and 120 elements about the structure. The angle around the cylinder, Theta (θ), is measured from the front stagnation point in a clockwise direction (Figure 14). The constant order approximation results in step function solutions, the linear approximation as a linear solution from one end of the element to the other, and the parabolic approximation results in parabolic behavior of the solution. The spikes in the parabolic solution are expected due to the discontinuity of the geometry slope as discussed in Chapter 4 and in Reference 8. Note that as the element size decreases, the amplitude of the spikes decreases also, although the ratio of element size to spike amplitude increases. If the geometry was modeled with slope continuity, it would be expected that the parabolic approximation solution spikes would smooth.

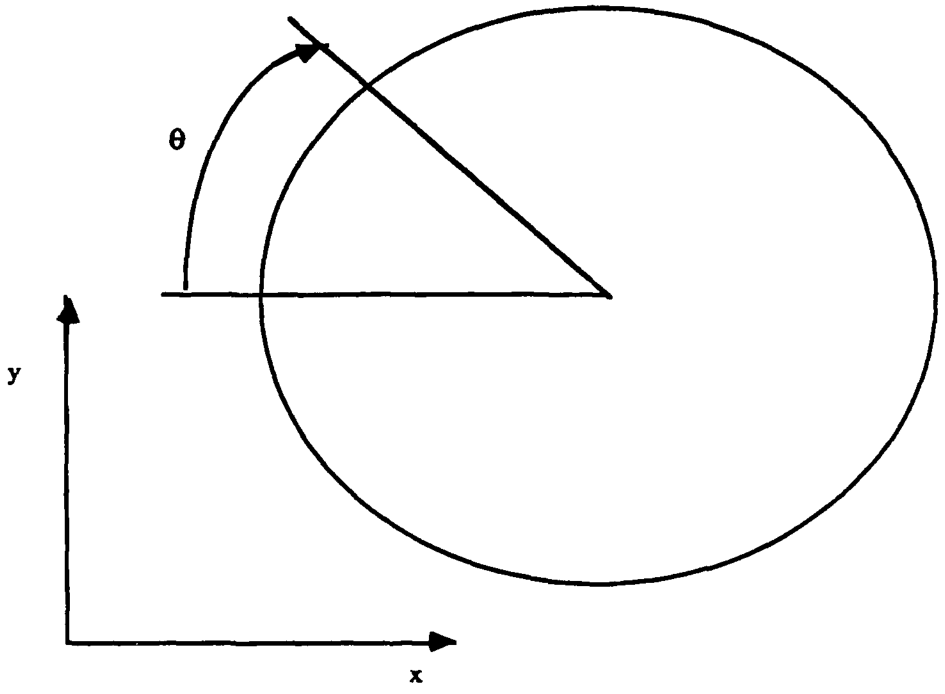


Figure 14. Definition of Angle Around Cylinder

Effect of Approximation Order on Solution

The approximations are compared against each other for 8, 40 and 120 elements in Figures A3, A4 and A5. Higher order approximations than parabolic would also display singular behavior near the corners.

Effect of End Node Location on Solution

The effect of moving the end nodes towards the element corner are shown in Figures A7 and A8. Parabolic approximating order was used for this comparison. The location of the end nodes does appear to effect the solution. As shown, moving the nodes closer to the corners causes the spiking behavior to amplify. This is due to the ln(1/r) and 1/r singular behavior of the fundamental solutions which cause the integrals to increase towards infinity. Indeed, when the nodes were placed within .1 δ of the corners, the solution quickly became invalid.

Comparison of Potential Flow Results With Theory

Linear approximations with 120 elements were chosen for the comparison of the BEM results with theory. Figures A9 through A12 show comparisons of the potential flow velocities and Cp calculations. For an fluid flow velocity of 1, theoretical values of Cp are 1 at the front and back stagnation points ($\theta = 0, 180$), and -3 as the flow acclerates around the top and bottom of the cylinder ($\theta = 90, 270$). As shown, the results of the code model the stagnation pressure well with any element size. The improvement of the solution due to decreasing element size is most apparent at the top and bottom of the cylinder as the solution converges towards theory. The largest error of the BEM appears at the top and bottom of the structure and is on the order of 2.4% of Cp. Figure A13, which displays the comparison of BEM potential with theory, shows that the BEM results overlay those of theory.

Effect of Number of Elements and Approximation Order on Solution Time

The elastic deformations were calculated using 40 flat elements and a linear approximation of the potential and deformation. Although 120 elements produced a slightly smoother pressure distribution than 40 elements, it also significantly increases the computer time required for solution. This is because a (8xm) x (8xm) system of equations is solved to determine the deformation. For 40 elements with linear modeling this produces a 640 x 640 system of equations which took approximately 40 minutes to execute on the AFIT Interim Computer Capability (ICC) system. The ICC is an Elxsi 6400 superminicomputer with the Berkeley 4.3 UNIX operating system. A 1920 x 1920 system of equations is produced for 120 elements with linear modeling. Due to this factor, it was felt that the solution with 40 elements was adequate to show trends and results of the BEM code. For flat elements and linear approximation of deformation, a larger number of elements would serve to slightly smooth the final results.

Elastic Deformation Results

The density of the potential fluid was defined as that of water at standard day temperature (1.937 slugs/ft^3). The structure was modeled with a Young's modulus of 1000, and a Poisson's ratio of 0.3. These assignments were made so that the resulting deformations would be large and more apparent.

The x and y components of deformation about the outer boundary are shown in Figures A14 and A15. The figures represent the structure being deformed inward on the front and back sides, and pulled outward at the top and bottom (Figure 15). This outward pulling is caused by the accelerated flow producing negative values of Cp, and therefore the pressure at the top and bottom is outward. The deformation along the inner boundary, although not shown, was outward in all directions, nearly equally opposing the outer boundary on the front and back and pushing outward in the same amount as the outer boundary on the top and bottom (Figure 15). The inflections of the deformation curves are due to the element bulging.

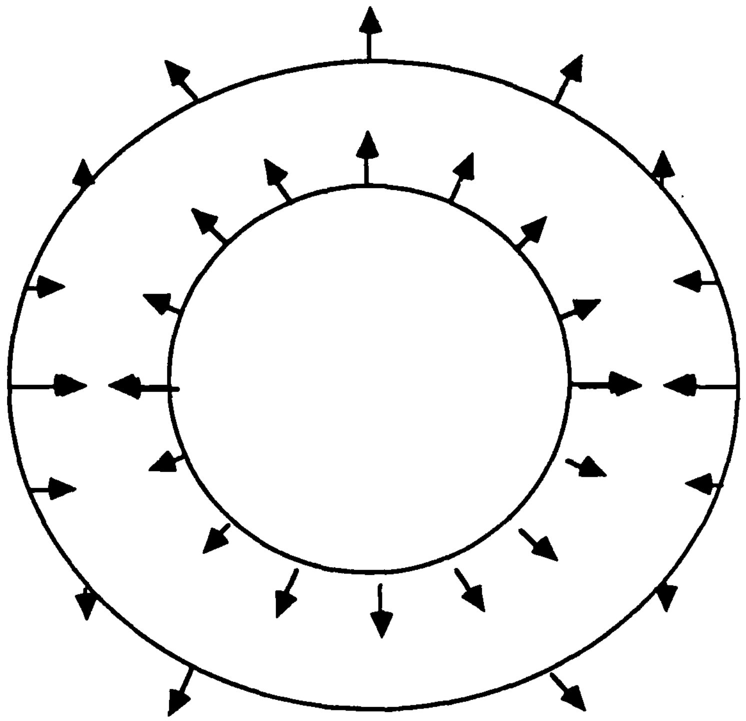


Figure 15. Direction of Deformations on the Structure

Rigid-Body Constraint

Note that although the structure was not physically pinned, rigid-body movement is constrained and the structure does not translate or rotate. There is zero deformation in the x direction at the top and bottom restricting side-to-side movement, and zero deformation in the y direction at the sides retricting up and down movement. This is due to the method of calculating the coefficient c^i with consideration of rigid-body movements. Effectively, by calculating the diagonal terms of the **H** matrix from a summation of the terms on the row, the solution is constrained. Additional boundary conditions may be placed on the structure if desired, and the system of equations treated accordingly.

Comparison With Finite Element Model

A finite element mesh was constructed with 40 linear elements. The internal pressure calculated from the BEM potential flow code was distributed at all the internal nodes. Concentrated forces were placed at the outer boundary nodes to represent the pressure distribution determined from the BEM solution.

Because the FEM nodes were at the corners of the elements, the BEM pressure was calculated at the end points using shape functions and averaged between elements to determine the concentrated pressure values at the corners.

The deformations from the FEM code are shown in Figures A16 and A17. The shape of the BEM and FEM solutions are nearly identical, and although the magnitudes are different, they are consistent with past experience of the FEM code used.

Iteration of Fluid/Structure Solutions

A linking geometry routine was written to apply the elastic deformations to the original structure boundary, and then to recalculate the potential and structural solutions. Once the original geometry was defined, no additional information was required from the user to iterate on the solutions. This factor is the major advantage to a boundary element formulation over the domain formulations as discussed in this research.

For the iterative results, 8 elements were used with linear approximation functions. The first calculated x and y components of deformation are plotted in Figures A18 and A20. Results from the third to the 9th iterations are shown in Figures A19 and A21. As shown, the magnitudes of the deformations decrease with each interation, and have approached zero within 12 decimal places by the 9th iteration thereby indicating a steady-state solution has been obtained.

VI. Conclusions and Recommendations

The major objective of this investigation was to determine the application of the boundary element technique to compute the incremental deformations of an elastic structure in fluid flow. The main advantage to the boundary element formulation over finite differences or finite elements is that it can numerically approximate both the fluid and elastic structure governing equations with a common definition of the fluid/structure boundary.

Results obtained during the course of this research indicate the the BEM does have a favorable application to the aeroelastic problem of fluid flow past an elastic body. While simplifying assumptions have been made in the execution of this thesis, it could be seen that this method presents a user friendly approach to the problem definition. The fact that once the original boundary is specified the user is not required to continuously redefine the geometry would be a large advantage in using organizations such as the 4950th Test Wing. Also, as implemented in this study, the user is able to use several different approximating orders rather than be limited to one. This would allow better modeling of the solution as a function of the problem to be solved.

As shown with the results from the program, the BEM can provide accurate results. The iterative capability of the code was also verified. This capability should prove to be an improvement over current computational techniques of the calculation of airloads and deformations in an aeroelastic system. While the process did produce steady-state solutions, the stability of the BEM iterative formulation should be more thoroughly researched. Stability criteria, such as that used in finite difference and finite element codes, may become applicable.

The largest factor towards accurate implementation of the formulation is the approximation of the boundary geometry. While presenting a simple method of implementation, discontinuous geometry slope produces many undesirable results and the use of continuous higher order geometry elements is highly recommended for future researchers.

Several additional features could be added to increase the versitility of the code. For example;

three-dimensional problems, adding wakes to properly solve for lifting body problems, and using non-stead; problem implementation. The applicability of the FEM to aeroelastic problems would be increased by each of these additions.

Appendix A: Investigation Results

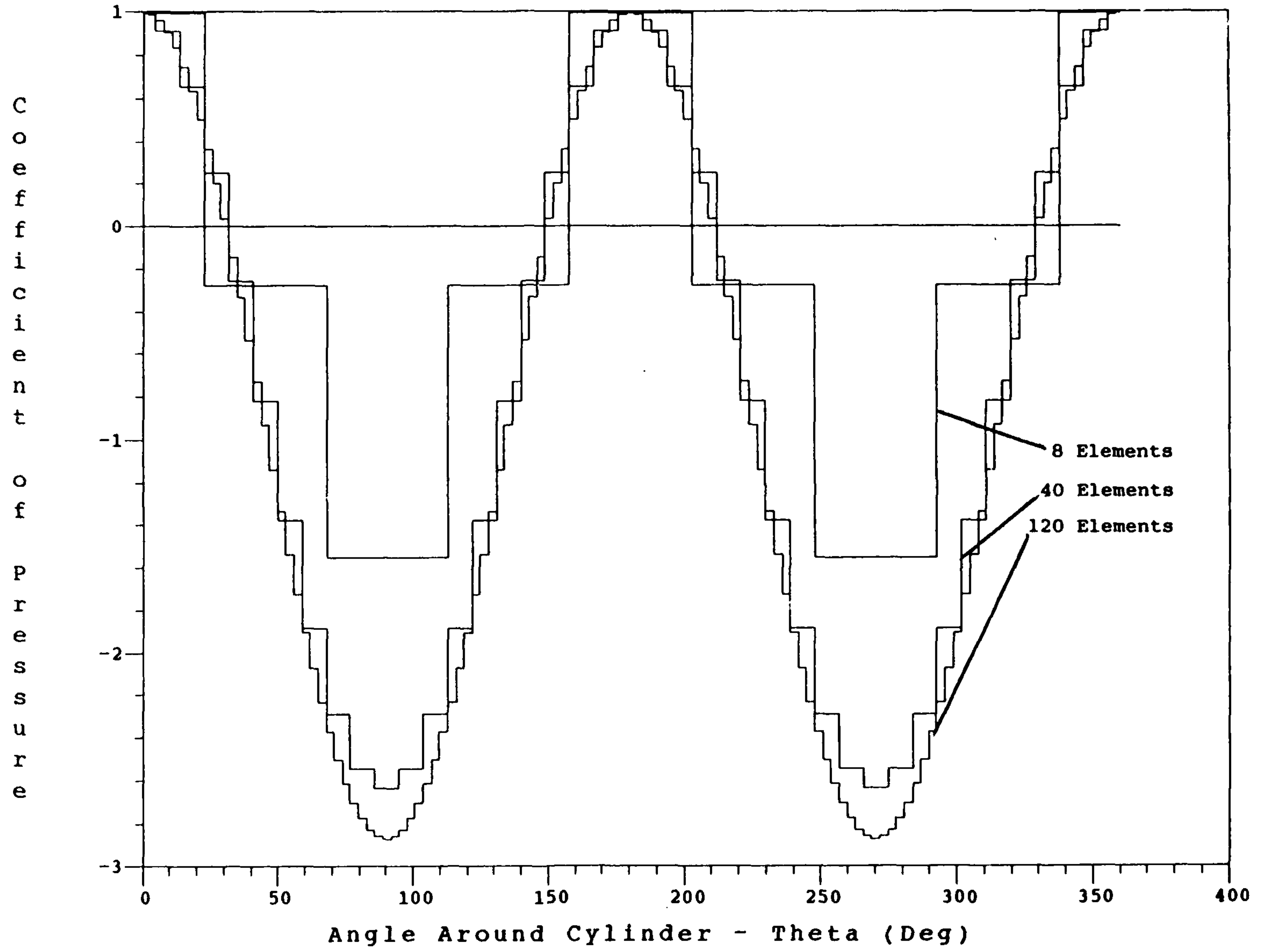


Figure A1. Effect of Element Size on Solution, Constant Modeling of Cp

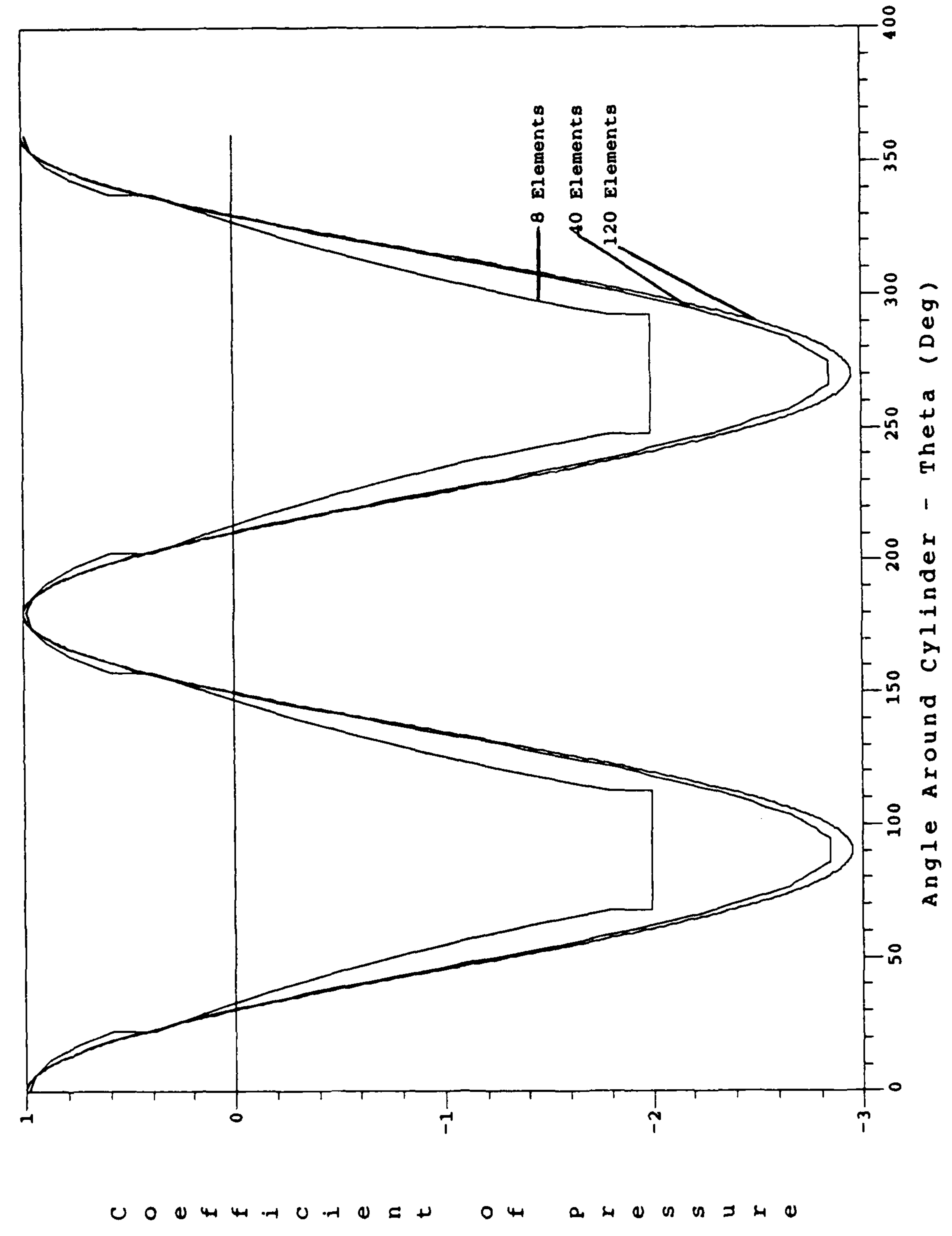


Figure A2. Effect of Element Size on Solution, Linear Modeling of Cp

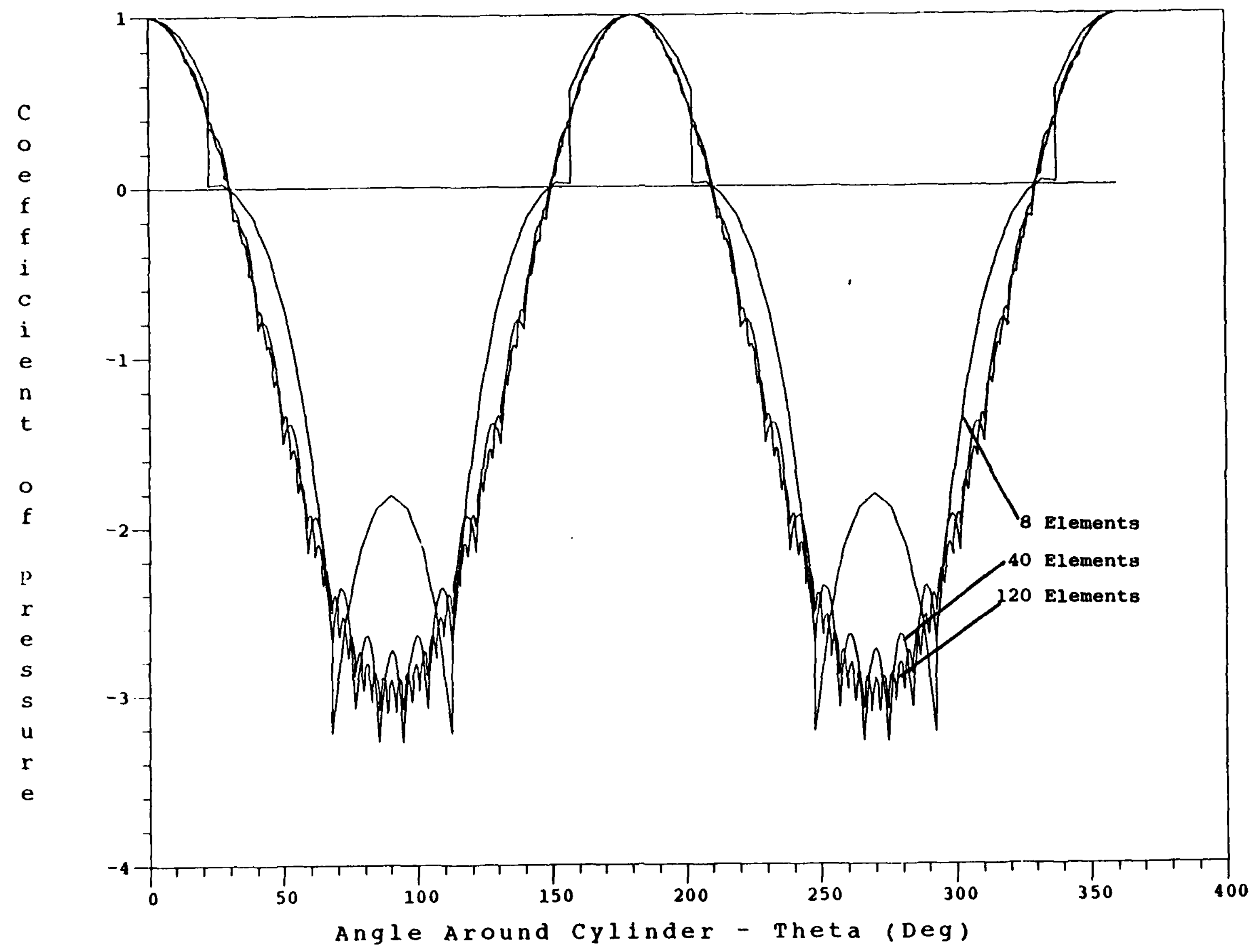


Figure A3. Effect of Element Size on Solution, Parabolic Modeling of Cp

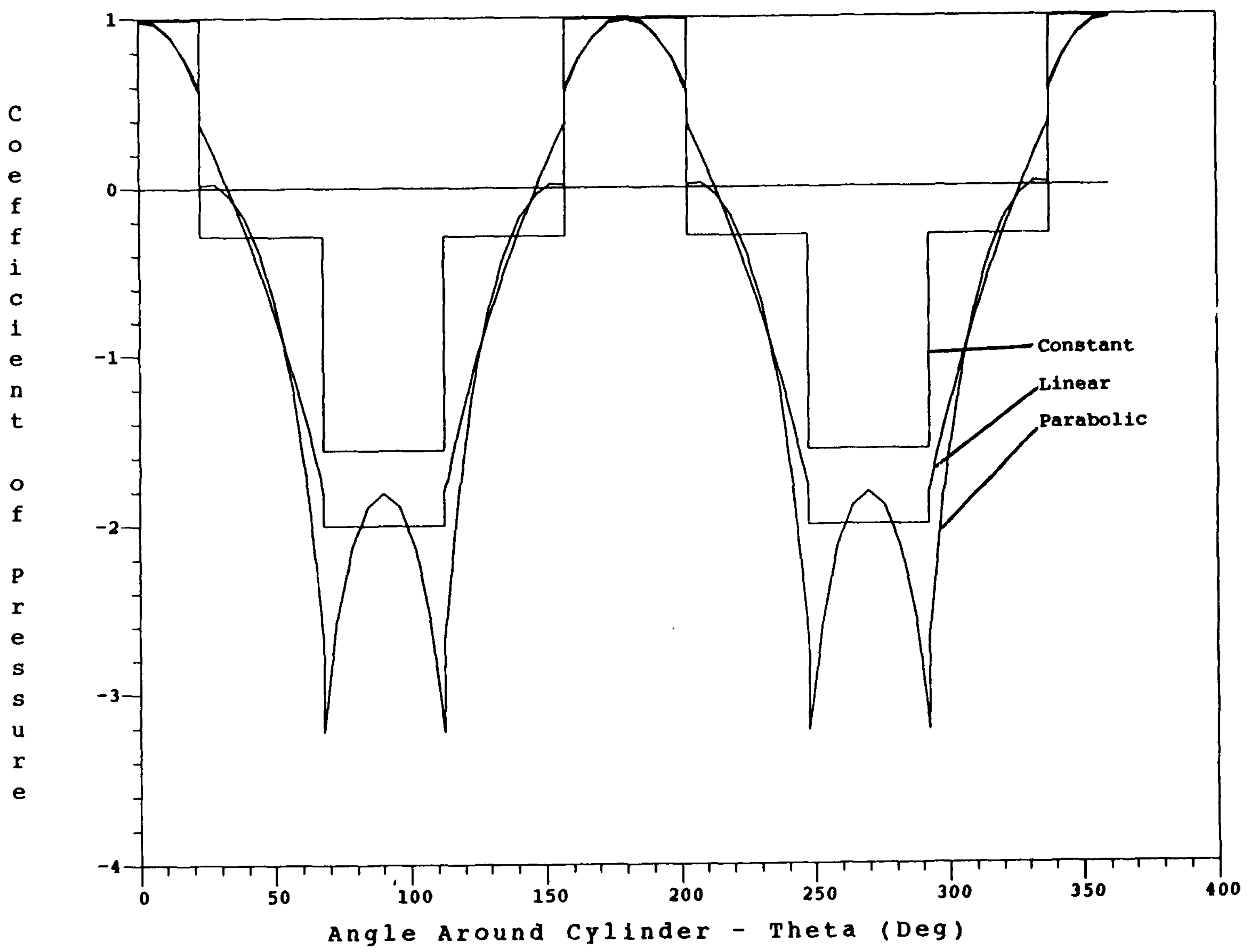


Figure A4. Effect of Function Approximation on Solution, 8 Elements

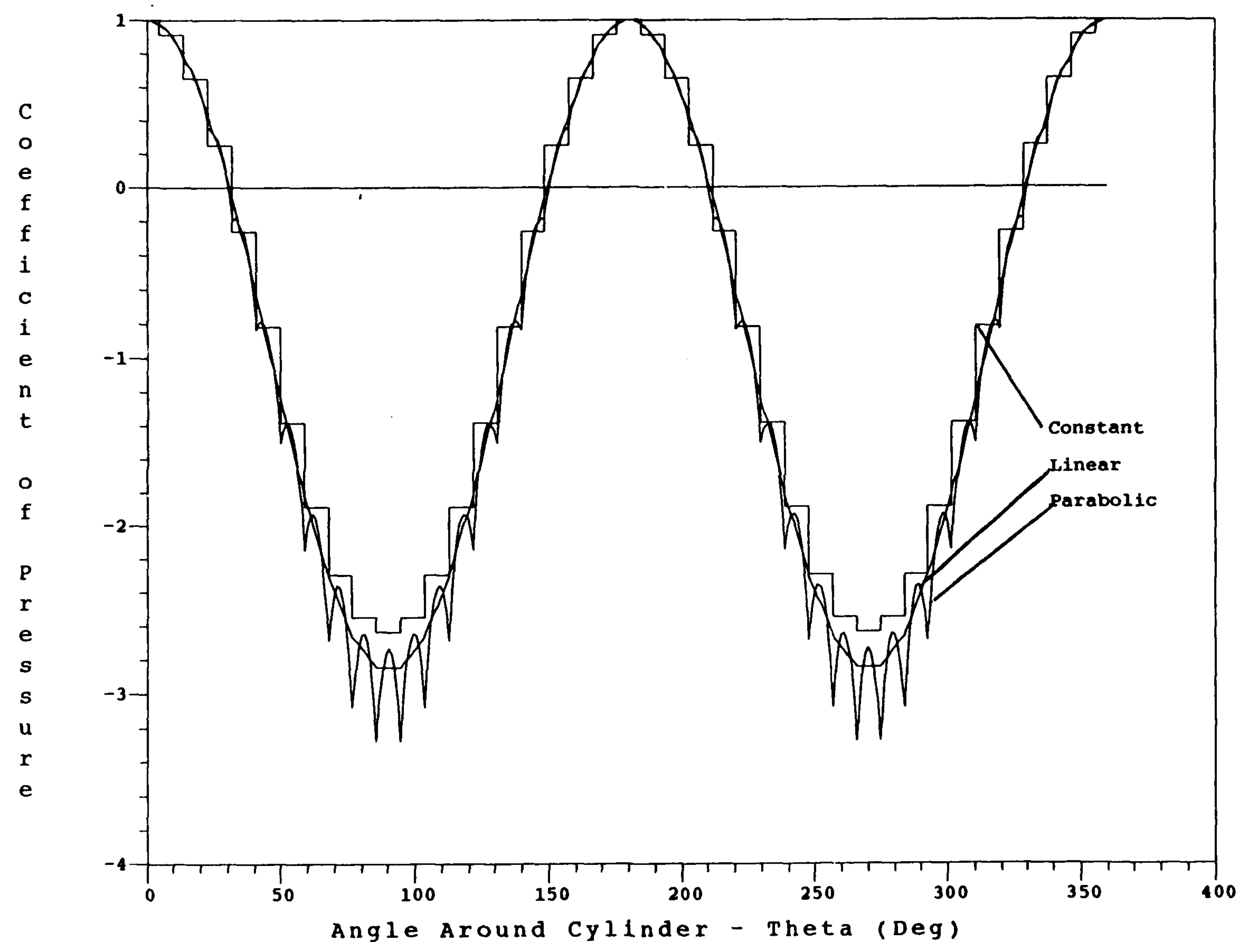


Figure A5. Effect of Function Approximation on Solution, 40 Elements

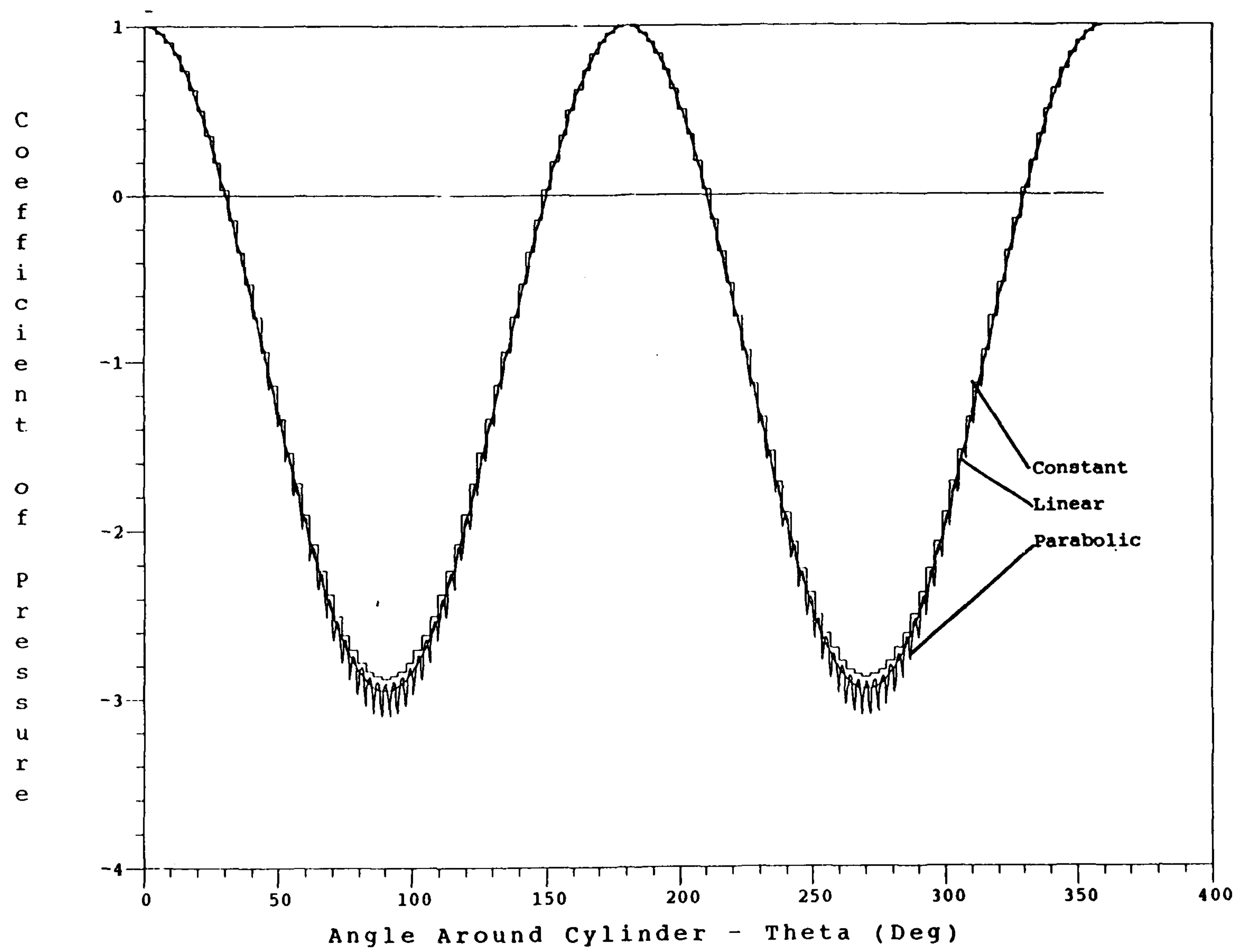


Figure A6. Effect of Function Approximation on Solution, 120 Elements

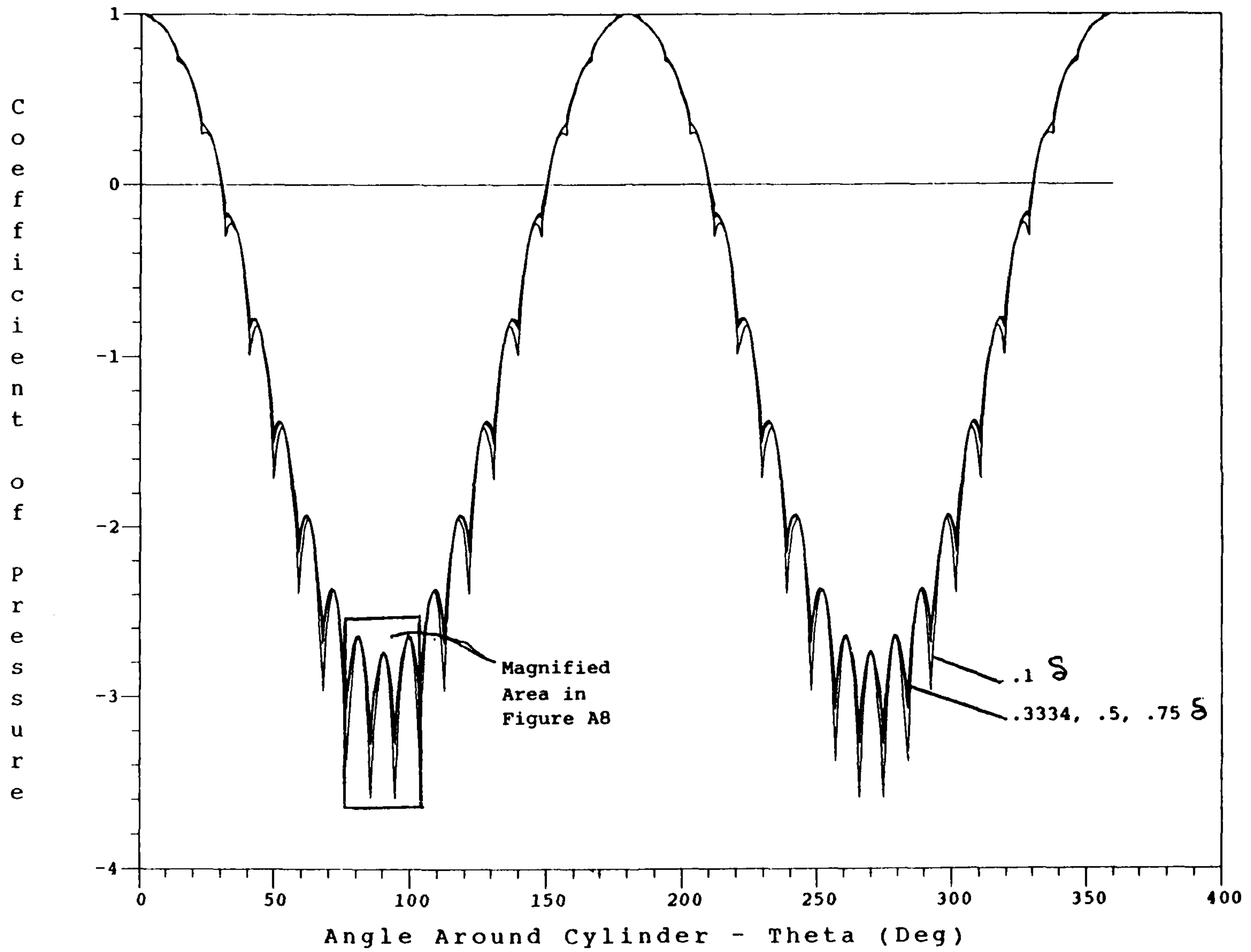


Figure A7. Effect of End Node Location on Solution, 40 Elements, Parabolic Approximation of Cp

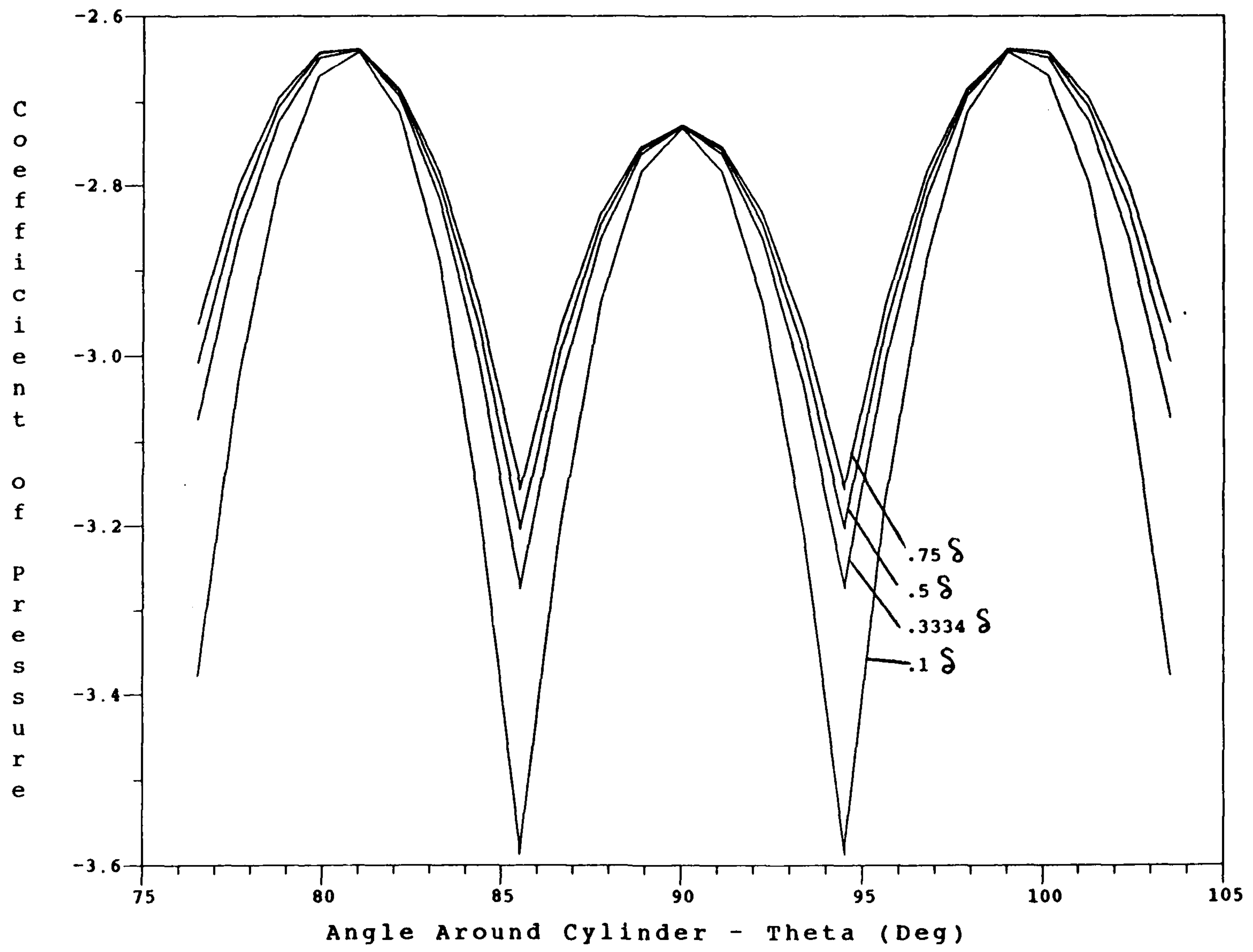


Figure A8. Effect of End Node Location on Solution, 40 Elements, Parabolic Approximation of Cp (Magnified)

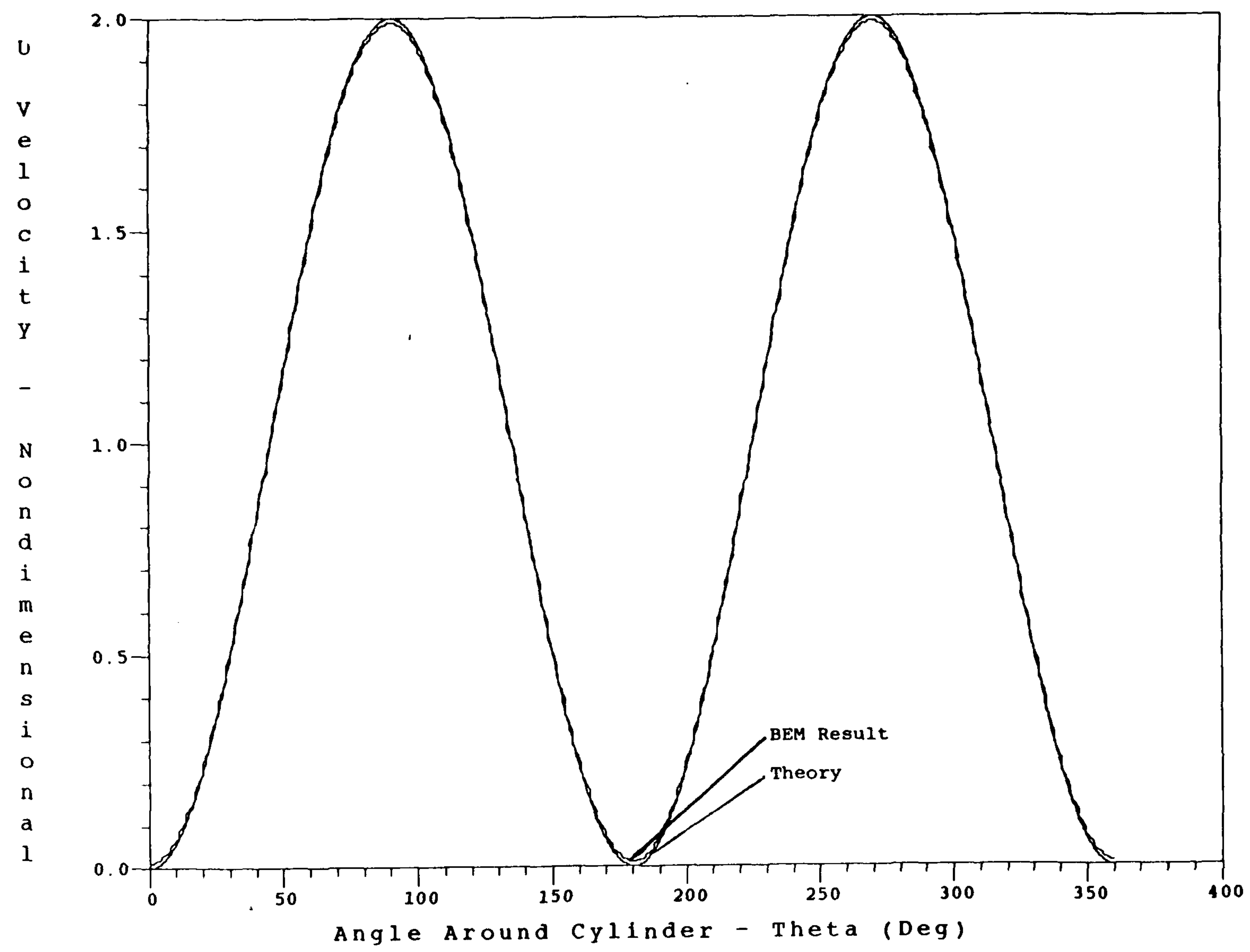


Figure A9. Comparison of X Component of Velocity With Theory, 120 Elements, Linear Approximation

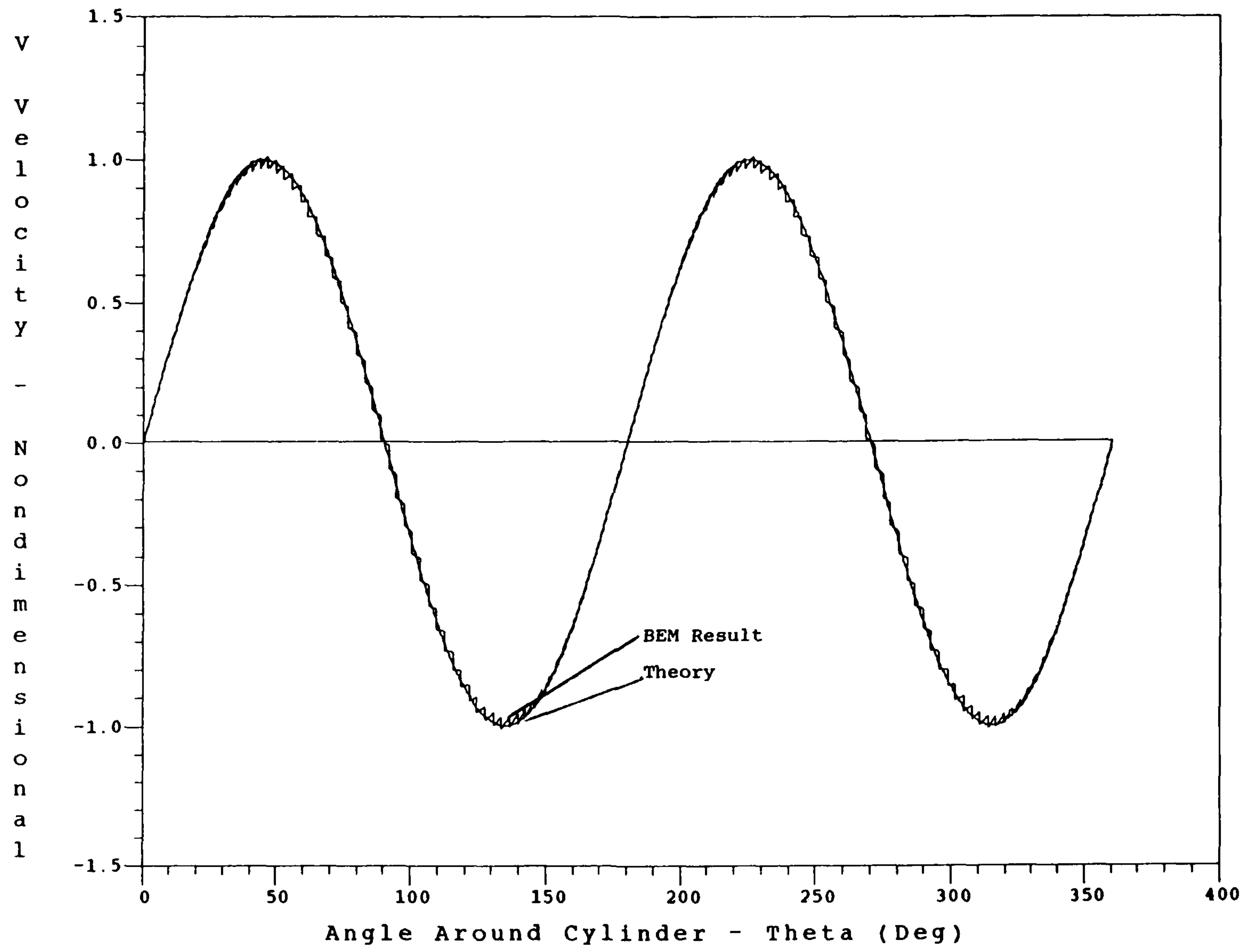


Figure A10. Comparison of Y Component of Velocity With Theory, 120 Elements, Linear Approximation

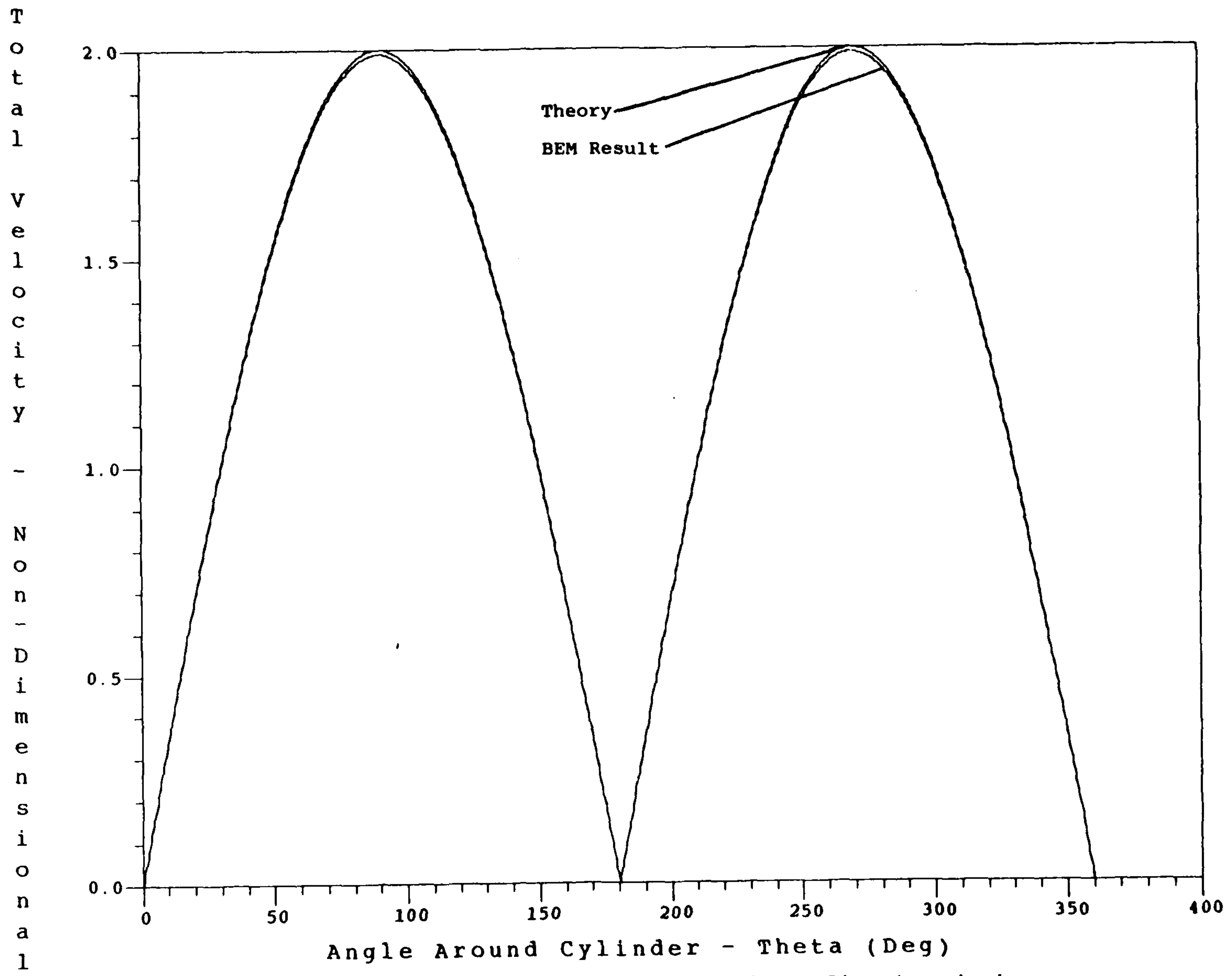


Figure A11. Comparison of Velocity With Theory, 120 Elements, Linear Approximation

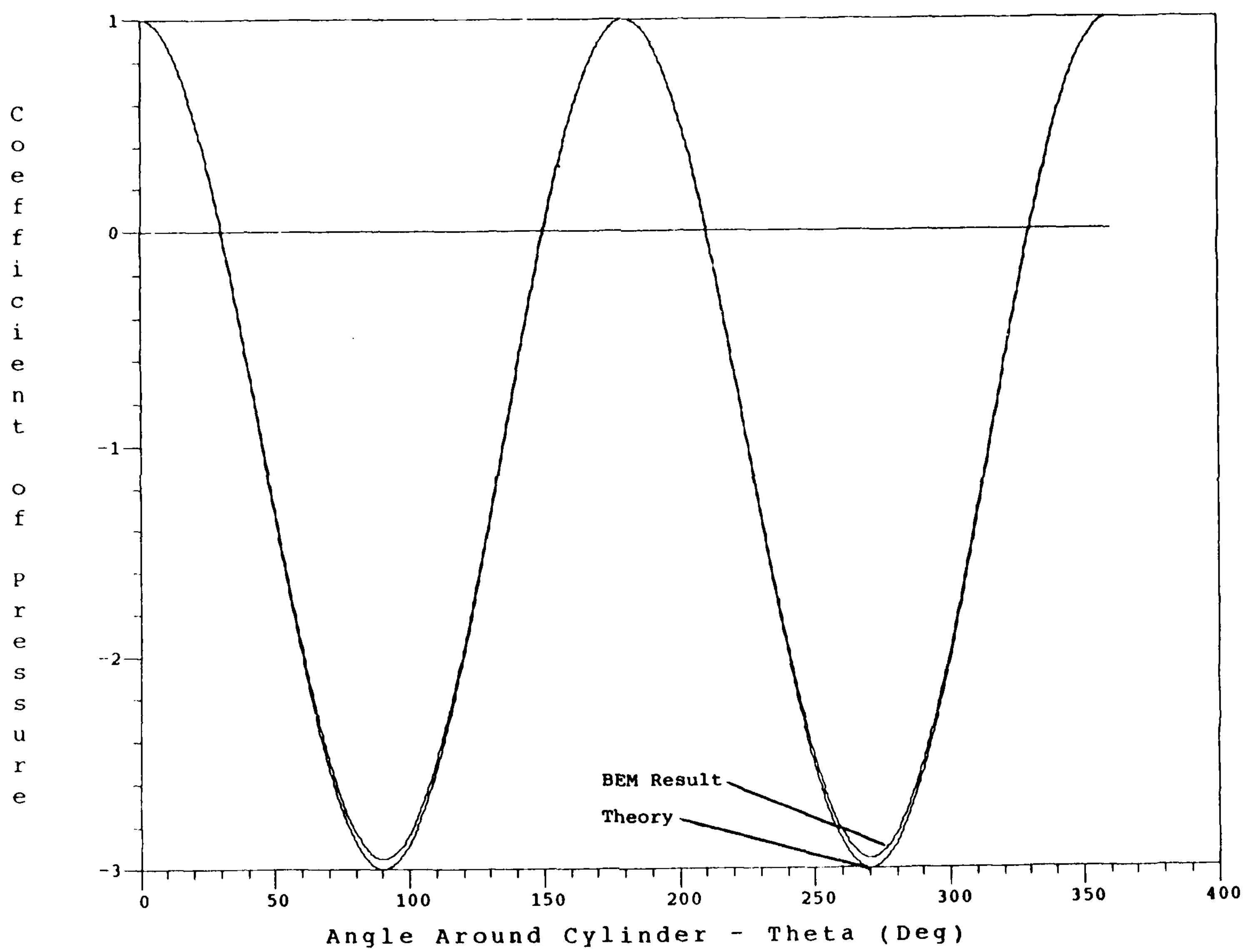


Figure A12. Comparison of Coefficient of Pressure With Theory, 120 Elements, Linear Approximation

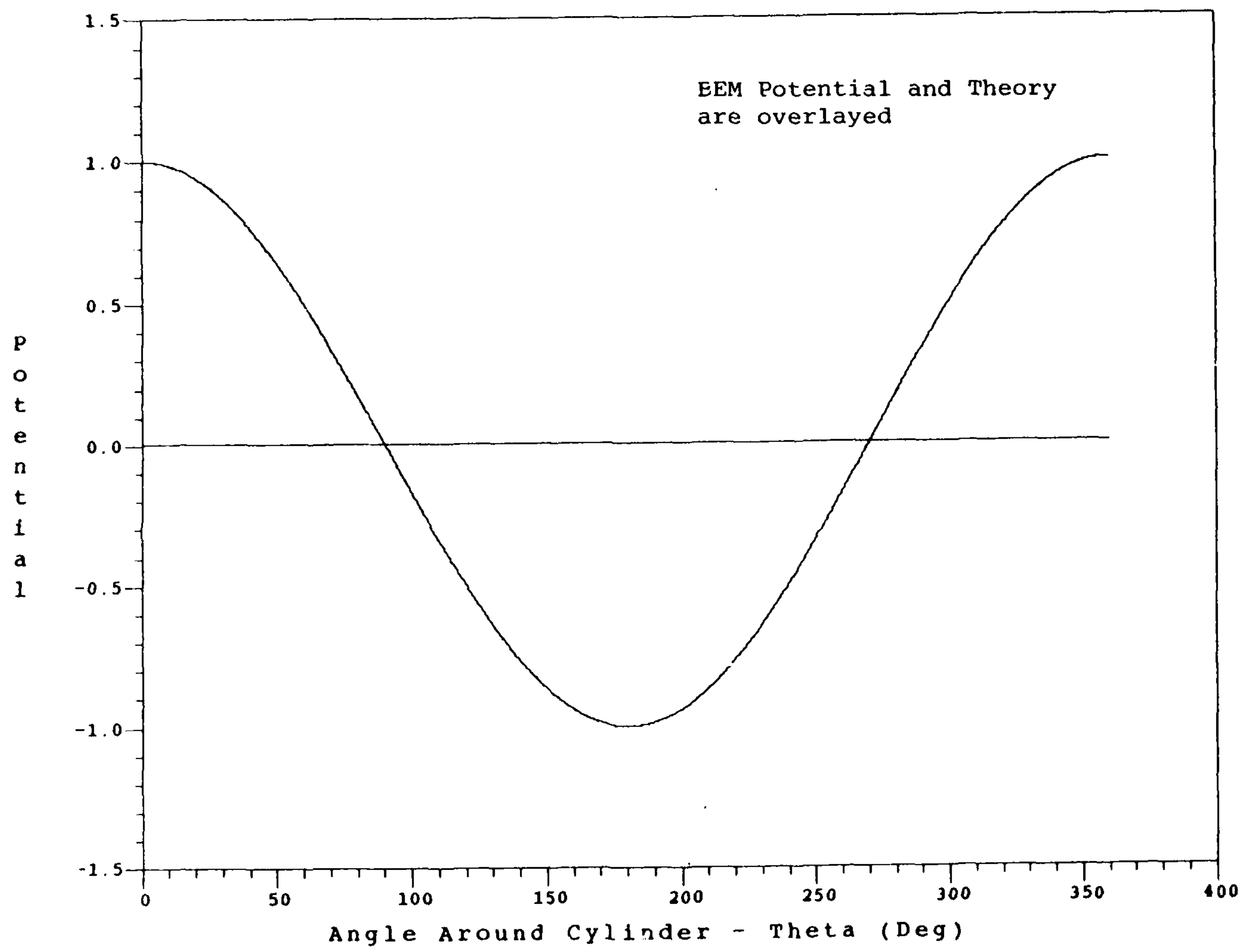


Figure A13. Comparison of Potential With Theory, 120 Element, Linear Approximation

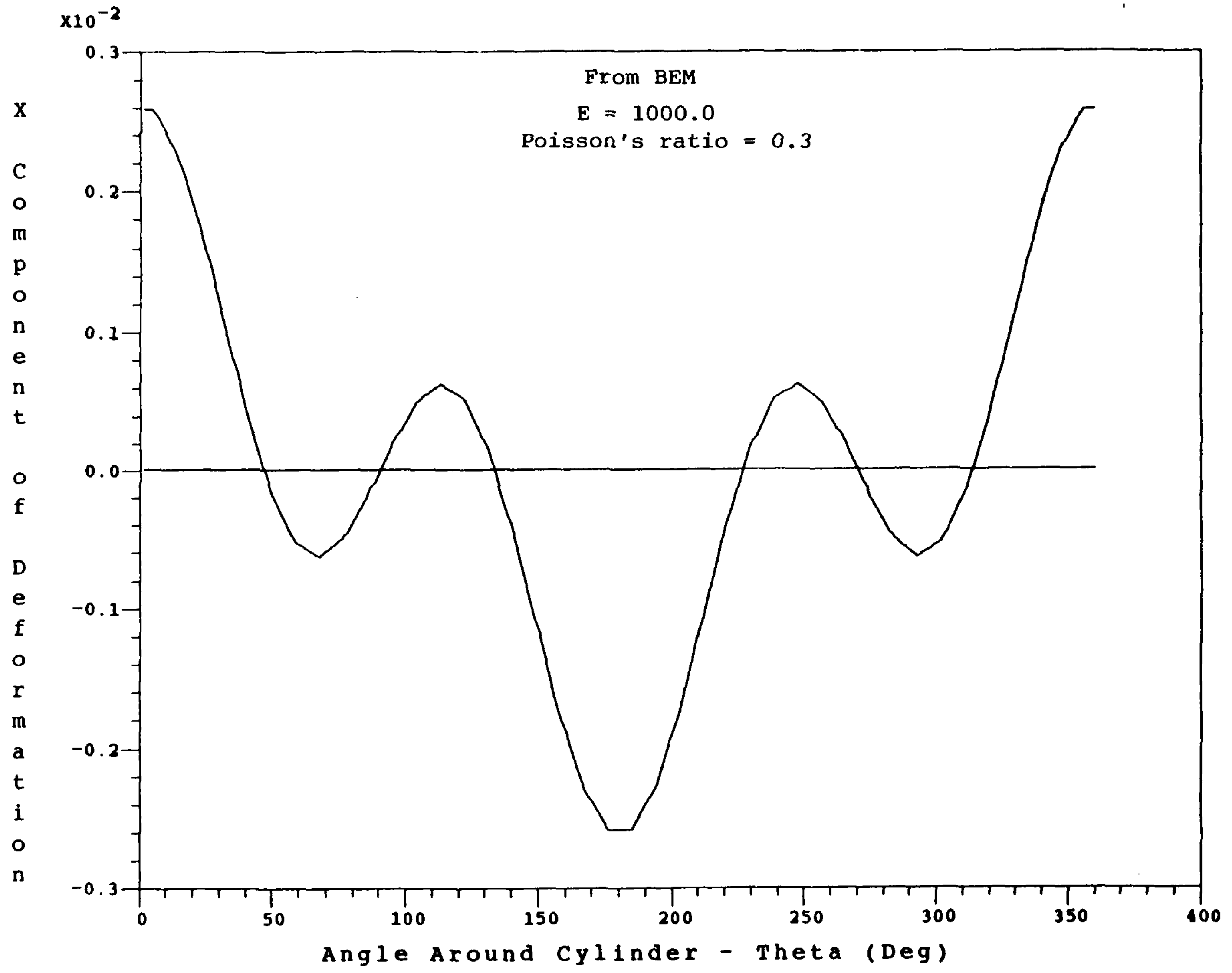


Figure A14. X Component of Deformation, 40 Elements, Linear Approximation

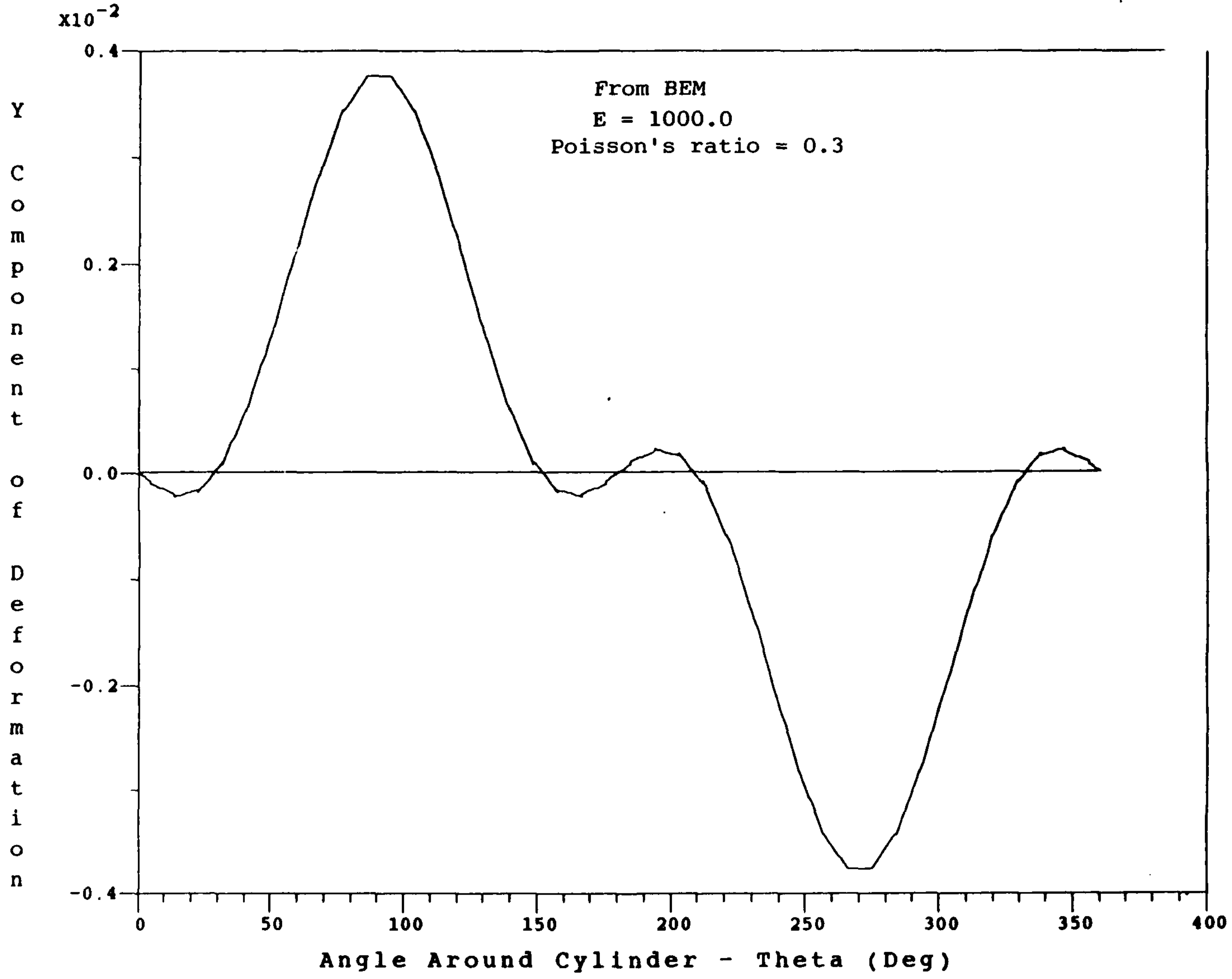


Figure A15. Y Component of Deformation, 40 Elements, Linear Approximation

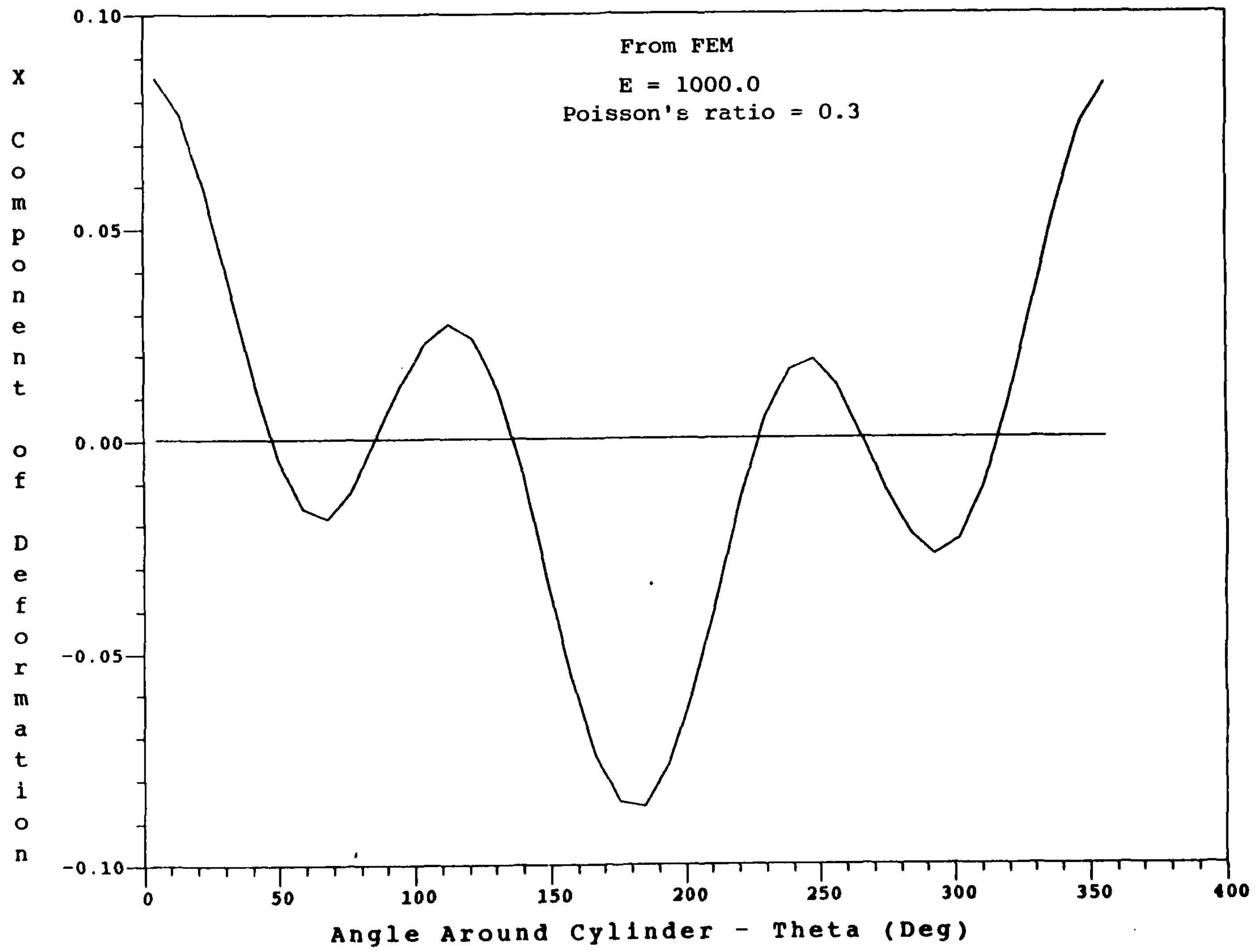


Figure A16. X Component of Deformation from FEM, 40 Elements, Linear Approximation

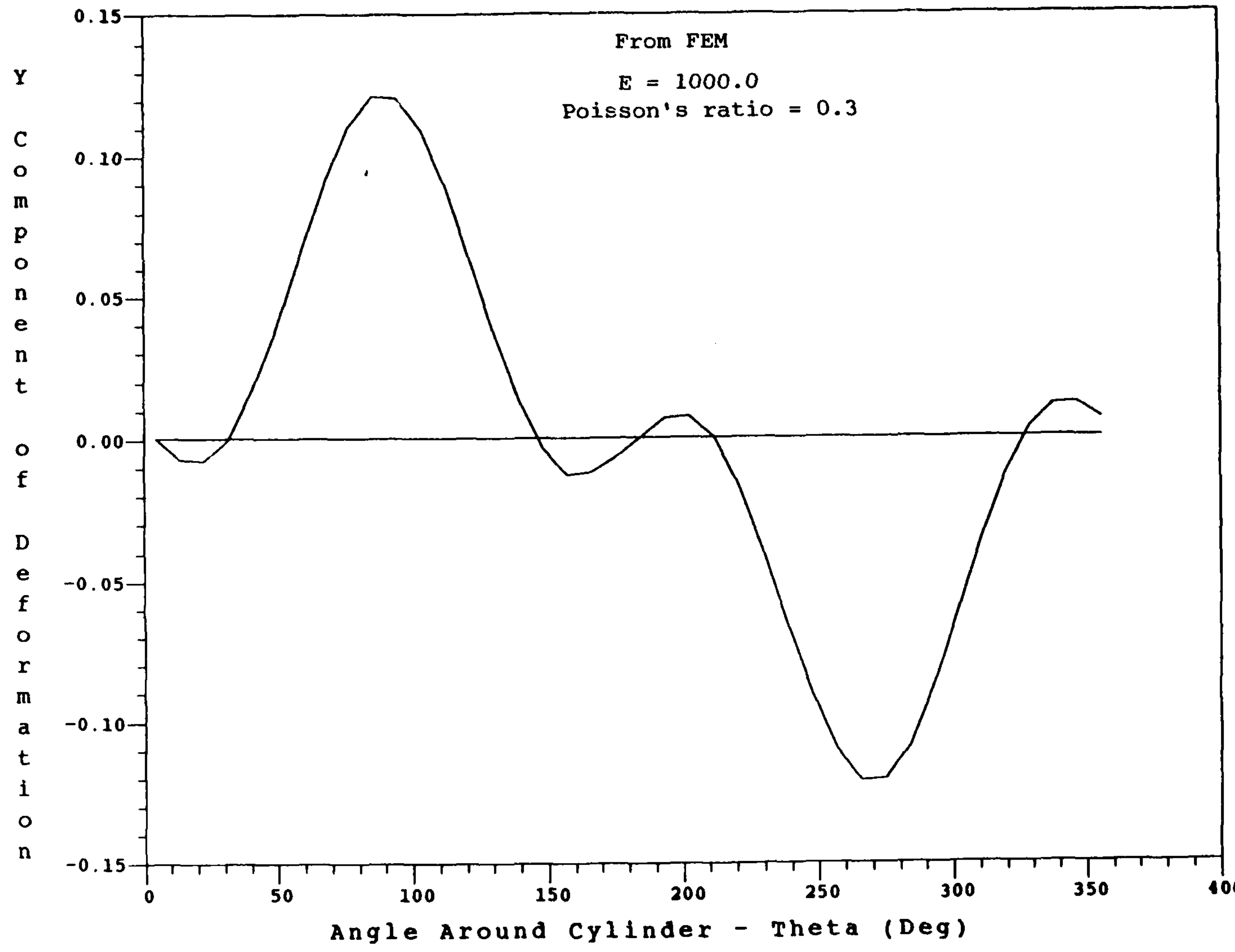


Figure A17. Y Component of Deformation From FEM, 40 Elements, Linear Approximation

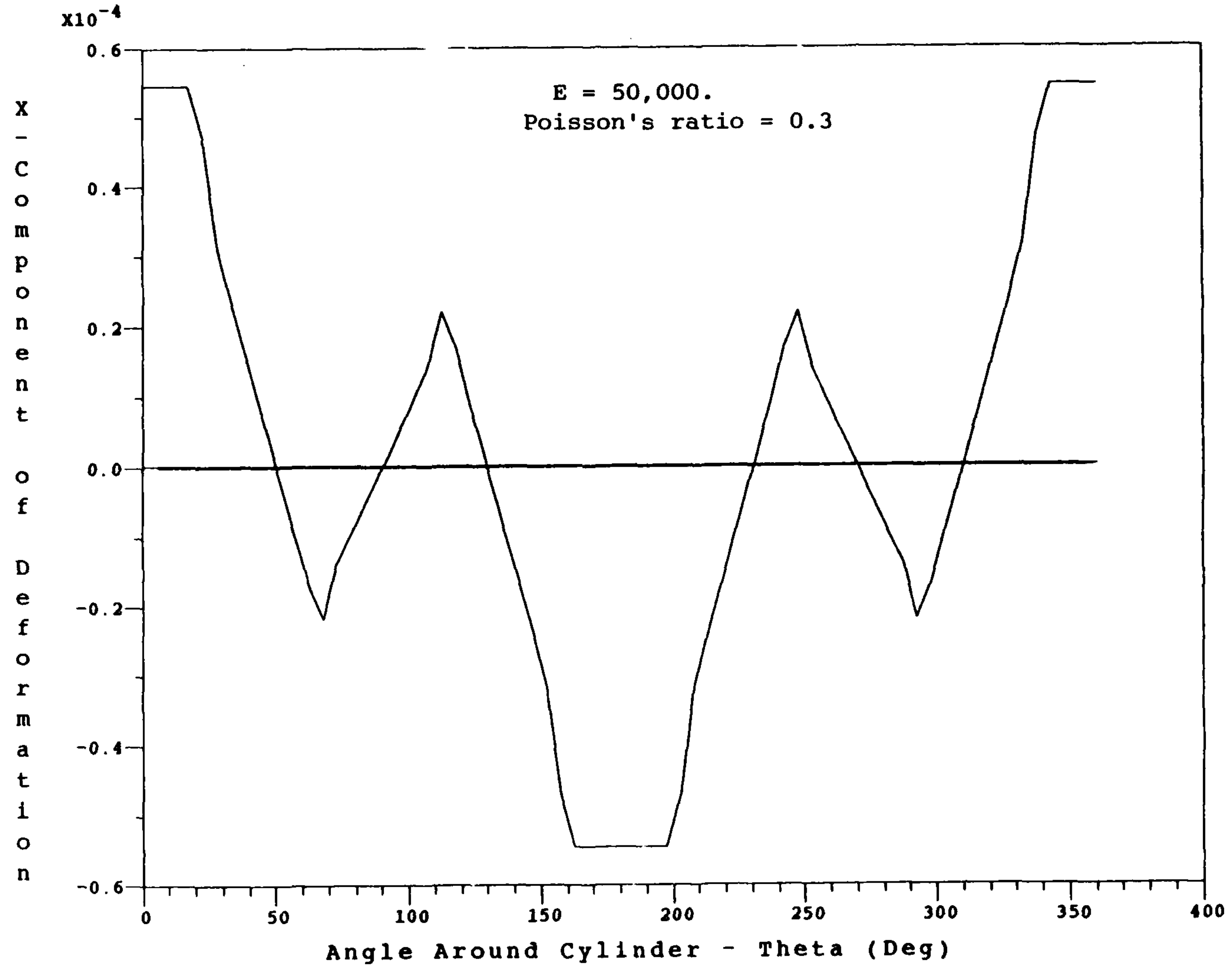


Figure A18. X Component of Deformation After 1 Pass, 8 Elements, Linear Approximation

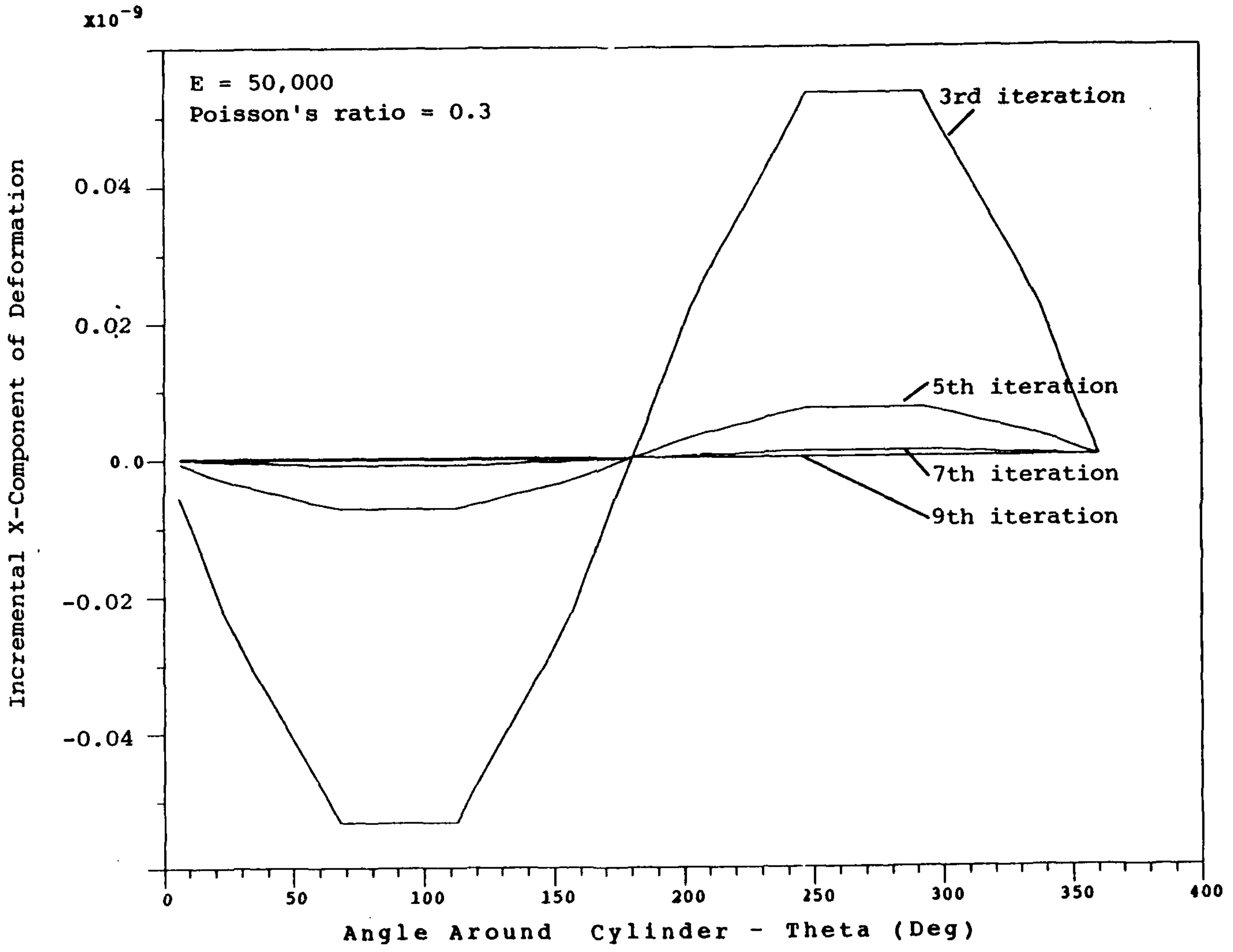


Figure A19. Iterative X Component of Deformation, 3-9 Iterations, 8 Elements, Linear Approximation

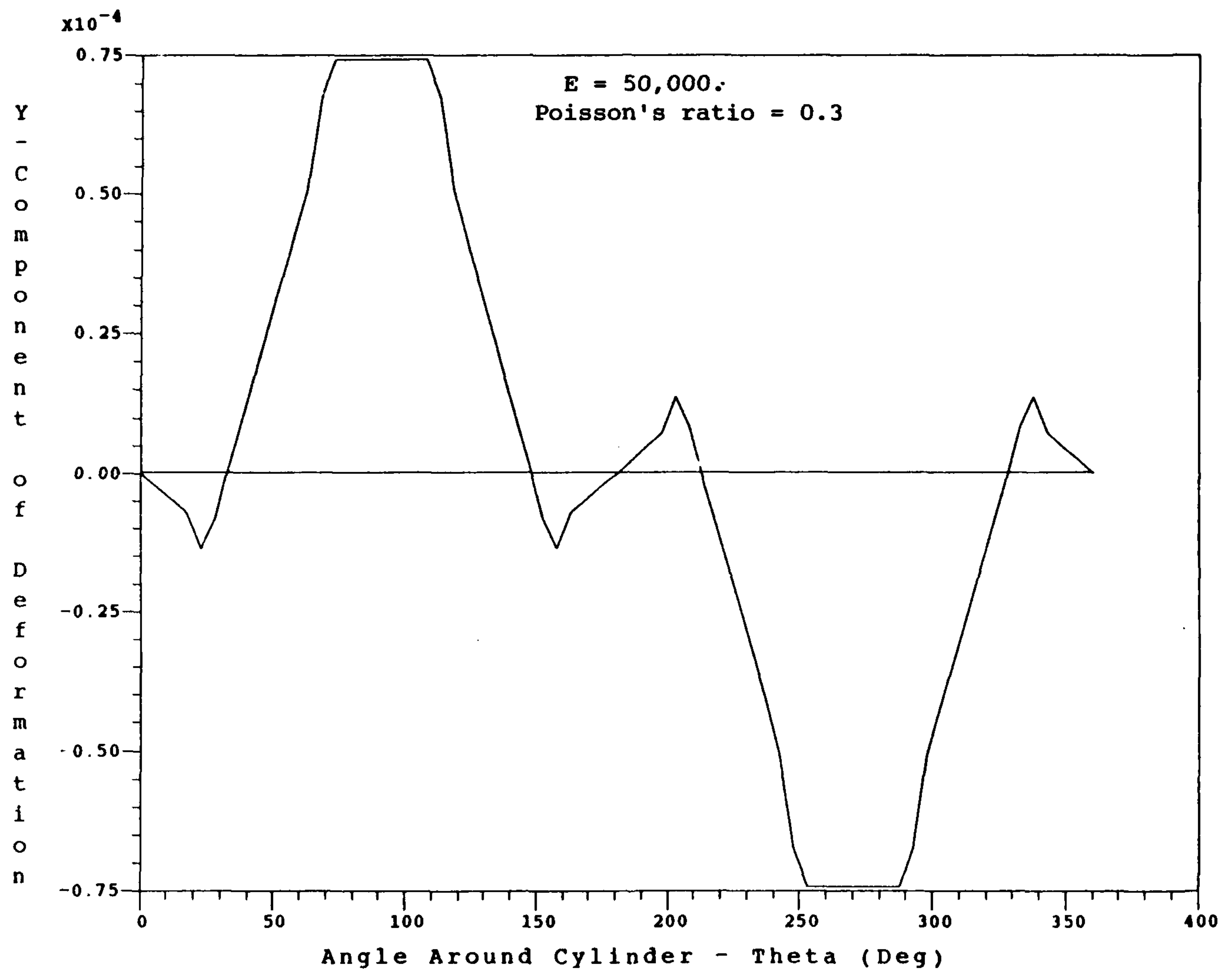


Figure A20. Y Component of Deformation After 1 Pass, 8 Elements, Linear Approximation

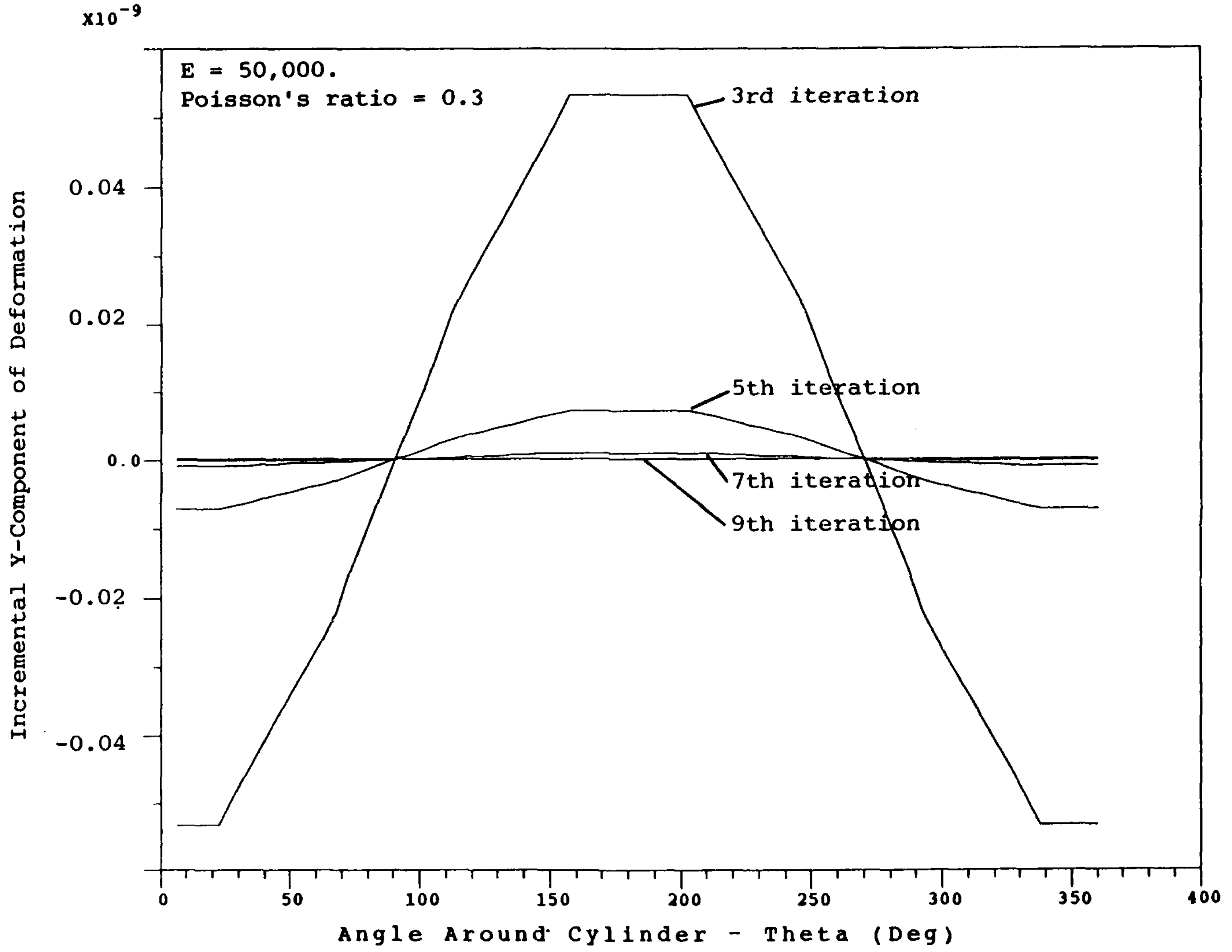


Figure A21. Iterative Y Component of Deformation, 3-9 Iterations, 8 Elements, Linear Approximation

Appendix B: Program Routines

```
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
C
C  FLUID - STRUCTURE INTERACTION USING BOUNDARY ELEMENT TECHNIQUE
C
C  THESIS PROJECT
C  NORMA F TAYLOR
C  GAE-88D
C
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
C
C  MAIN PROGRAM
C
C  IFLAG = 1  PERFORM FLUID CALCULATIONS ONLY
C        = OTHER  ITERATE BETWEEN FLUID AND STRUCTURE
C  PFLAG = 1  PRINT OUT POTENTIAL INFORMATION
C        = 2  PRINT OUT STRUCTURAL INFORMATION
C        = 3  PRINT OUT BOTH
C
C*******************************************************************
      implicit real*8  (a-h,o-z)
      COMMON/FLAGS/IFLAG,PFLAG,ITER
      COMMON/GEOMET/NNODES,NELEMS,ORDER,EPS,NORMI,NORMJ,JAC,NODEC,
     *NODPIN1,NODPIN2,NODE(5),ELEM(360,4)
      COMMON/POTEN/VINF,ALPHA,VI,VJ,PINF,RHOINF,GAMMA,MINF2,NODEP
      COMMON/FLOW/H(240,240),G(240,240),Q(240),GHAT(240)
      COMMON/STRUCT/GE,XNU,HS(1920,1920),GS(1920,1440),GSHAT(1920)
      COMMON/POSITION/XI(1080),YI(1080)
      COMMON/VELOCIT/UVEL(240),VVEL(240),XFORC(480),YFORC(480),
     *XOLD(480),YOLD(480)
      REAL*8  NODE,GAMMA,MINF2,NORMI,NORMJ,JAC
      INTEGER ORDER,PFLAG
C
C
C
      READ(5,1) IFLAG,PFLAG
 1    FORMAT(2I1)
      READ(5,2) ITERATE
 2    FORMAT(I2)
C
      IF(IFLAG.EQ.1) THEN
C
C  PERFORM FLUID CALCULATIONS ONLY
C
C  READ IN GEOMETRY AND FLUID INFORMATION
C  FORM H(PHI)=GHAT SYSTEM
C  SOLVE FOR PHI
C  CALCULATE PRESSURES (XFORC,YFORC)
C
        CALL GEOMIN
        CALL GOUTPUT(ITER)
        CALL FLUIDIN
        CALL FFORM
        CALL QFLOW
        DO 5 I=1,NNODES
```

```
          GHAT(I)=0.0
          DO 5 J=1,NNODES
            GHAT(I)=GHAT(I)+G(I,J)*Q(J)
 5      CONTINUE
        CALL HDIAG
        IF(PFLAG.EQ.1.OR.PFLAG.EQ.3) CALL HGQOUT
        CALL SOLVE(G,GHAT,D,NNODES,240)
C
C  NOTE THAT AFTER GOING THROUGH SOLVE GHAT IS THE SOLUTION PHI
C
        CALL VELOC
        CALL PRESSURE
        CALL FLOWOUT
        CALL DRAW(ITERATE)
      ELSE
C
C  CALCULATE BOTH FLUID AND STRUCTURAL INFORMATION
C  AND ITERATE UNTIL STEADY STATE SOLUTION IS REACHED
C
C  READ IN GEOMETRY, FLUID AND STRUCTURAL INFORMATION
C  CALCULATE H(PHI)=GHAT SYSTEM
C  SOLVE FOR PHI
C  CALCULATE SURFACE PRESSURES
C  CALCULATE HS(U)=GSHAT SYSTEM
C  SOLVE FOR DEFORMATIONS U
C  CALCULATE NEW GEOMETRY
C  ITERATE
C
        ITER=0
        CALL GEOMIN
        CALL GOUTPUT(ITER)
        CALL FLUIDIN
        CALL SINFO
 100    CALL FFORM
        CALL QFLOW
        DO 105 I=1,NNODES
          GHAT(I)=0.0
          DO 105  J=1,NNODES
            GHAT(I)=GHAT(I)+G(I,J)*Q(J)
 105    CONTINUE
        CALL HDIAG
        IF(PFLAG.EQ.1.OR.PFLAG.EQ.3) CALL HGQOUT
        CALL SOLVE(H,GHAT,D,NNODES,240)
C
C  NOTE THAT AFTER GOING THROUGH SOLVE GHAT IS REALLY THE SOLUTION PHI
C
        CALL VELOC
        CALL PRESSURE
        CALL FLOWOUT
        CALL SFORM
        CALL HSDIAG
        CALL GSHATT
        IF(PFLAG.EQ.2.OR.PFLAG.EQ.3)  CALL SOUT
        CALL SOLVE(HS,GSHAT,D,NNODES*8,1920)
```

```
C
C  NOTE THAT AFTER SOLVES GSHAT IS REALLY THE DEFORMATION U
C
         CALL SOUTPUT(ITER)
         ITER=ITER+1
         CALL DRAW(ITER)
         CALL DEFENDS
         CALL GOUTPUT(ITER)
         IF(ITER.LT.ITERATE) GO TO 100
       ENDIF
  150  CONTINUE
       END
```

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C  DEFENDS
C
C  THIS ROUTINE CALCULATES THE DEFORMATIONS AT THE ENDS OF THE ELEMENTS
C  AND DETERMINES THE NEW ELEMENT COORDINATES
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      SUBROUTINE DEFENDS
      IMPLICIT REAL*8  (A-H,O-Z)
      COMMON/GEOMET/NNODES,NELEMS,ORDER,EPS,NORMI,NORMJ,JAC,NODEC,
     *NODPIN1,NODPIN2,NODE(5),ELEM(360,4)
      COMMON/STRUCT/GE,XNU,HS(1920,1920),GS(1920,1440),GSHAT(1920)
      COMMON/POSITION/XI(1080),YI(1080)
      REAL*8 NODE,NORMI,NORMJ,JAC,SHAP(5),SHP(5)
      INTEGER ORDER
C
C  CALCULATE THE X AND Y DEFORMATION OF THE LEFT END OF ELEMENT J
C  BY AVERAGING THE DEFORMATION AT ETA=-1 E'EMENT J AND
C  ETA=1 ELEMENT J-1
C
      DO 20 J=1,2*NELEMS
        UXL=0.0
        UXR=0.0
        UYL=0.0
        UYR=0.0
        CALL SHAPE(ORDER,-1.D0,SHAP)
        CALL SHAPE(ORDER,1.D0,SHP)
        DO 10 L=1,ORDER
          JL=J*ORDER-(ORDER-L)
          JR=(J-1)*ORDER-(ORDER-L)
          IF(J.EQ.1) JR=NELEMS*ORDER-(ORDER-L)
          IF(J.EQ.NELEMS+1) JR=2*NELEMS*ORDER-(ORDER-L)
          UXL=UXL+SHAP(L)*GSHAT(JL)
          UXR=UXR+SHP(L)*GSHAT(JR)
          UYL=UYL+SHAP(L)*GSHAT(4*NNODES+JL)
          UYR=UYR+SHP(L)*GSHAT(4*NNODES+JR)
10      CONTINUE
        ELEM(J,1)=ELEM(J,1)+(UXL+UXR)/2.
        ELEM(J,2)=ELEM(J,2)+(UYL+UYR)/2.
20    CONTINUE
C
C  CALCULATE THE RIGHT ENDS OF THE ELEMENTS BASED ON THEM
C  EQUALING THE LEFT ENDS OF THE J+1 ELEMENT
C
      DO 30 J=1,2*NELEMS
        IF(J.EQ.NELEMS) THEN
          ELEM(J,3)=ELEM(1,1)
          ELEM(J,4)=ELEM(1,2)
        ELSE
          IF(J.EQ.2*NELEMS) THEN
            ELEM(J,3)=ELEM(NELEMS+1,1)
            ELEM(J,4)=ELEM(NELEMS+1,2)
          ELSE
            ELEM(J,3)=ELEM(J+1,1)
            ELEM(J,4)=ELEM(J+1,2)
          END IF
        END IF
30    CONTINUE
C
C  CALCULATE THE CONNECTING ELEMENT END POINTS
C
      DO 40 I=1,NELEMS
        ELEM(2*NELEMS+I,1)=ELEM(I,3)
        ELEM(2*NELEMS+I,2)=ELEM(I,4)
        ELEM(2*NELEMS+I,3)=ELEM(NELEMS+I,3)
        ELEM(2*NELEMS+I,4)=ELEM(NELEMS+I,4)
40    CONTINUE
C
C  CALCULATE THE NEW COORDINATES OF THE NODAL POINTS
C
      DO 50 I=1,NELEMS*3
        DO 50 L=1,ORDER
          ETAA=-1.
          ETAC=1.
          ETAQ=NODE(L)
          SHAP1=(ETAQ-ETAC)/(ETAA-ETAC)
          SHAP2=(ETAQ-ETAA)/(ETAC-ETAA)
          XI(I*ORDER-(ORDER-L))=SHAP1*ELEM(I,1)+SHAP2*ELEM(I,3)
          YI(I*ORDER-(ORDER-L))=SHAP1*ELEM(I,2)+SHAP2*ELEM(I,4)
50    CONTINUE
      RETURN
      END
```

```
C*****************************************************************
C
C  FFORM
C
C  THIS SUBROUTINE COMPUTES THE H AND G MATRICES FOR THE
C  POTENTIAL FLOW
C
C  H MATRIX = INTEGRAL(SHAPE*QSTAR*JAC)DETA
C
C  G MATRIX = INTEGRAL(SHAPE*PHISTAR*JAC)DETA
C
C*****************************************************************
      SUBROUTINE FFORM
      implicit real*8  (a-h,o-z)
      COMMON/POSITION/XI(1080),YI(1080)
      COMMON/GEOMET/NNODES,NELEMS,ORDER,EPS,NORMI,NORMJ,JAC,NODEC,
     *NODPIN1,NODPIN2,NODE(5),ELEM(360,4)
      COMMON/FLOW/H(240,240),G(240,240),Q(240),GHAT(240)
      real*8 ETA(12),WT(12),ETALN(10),WTLN(10),ETA2(10),WT2(10)
      real*8 SHAP(5),SHPLN2(5),SHPLN(5),SHAP3(5)
      real*8 NODE,JAC,JACLN,JACLN2,JAC3,JAC4,NORMI,NORMJ
      INTEGER ORDER
C
C     ETA AND WT ARE NORMAL 12TH ORDER GAUSSIAN QUADRATURE
C     VALUES AND WEIGHTS
C
      DATA ETA(1),ETA(2),ETA(3),ETA(4),ETA(5),ETA(6),ETA(7),
     *ETA(8),ETA(9),ETA(10),ETA(11),ETA(12)/-.98156063d0,-.90411726d0,
     *-.76990267d0,-.58731795d0,-.36783149d0,-.12523341d0,
     *.12523341d0,.36783149d0,.58731795d0,.76990267d0,.90411726d0,
     *.98156063d0/
      DATA WT(1),WT(2),WT(3),WT(4),WT(5),WT(6),WT(7),WT(8),
     *WT(9),WT(10),WT(11,,WT(12)/.04717534d0,.10693933d0,.16007833d0,
     *.20316743d0,.23349254d0,.24914705d0,.24914705d0,.23349254d0,
     *.20316743d0,.16007833d0,.10693933d0,.04717534d0/
C
C     ETALN AND WTLN ARE 10TH ORDER LOGARITHMIC GAUSSIAN QUADRATURE
C     VALUES AND WEIGHTS
C
      DATA ETALN(1),ETALN(2),ETALN(3),ETALN(4),ETALN(5),ETALN(6),
     *ETALN(7),ETALN(8),ETALN(9),ETALN(10)/.00904259d0,.05397105d0,
     *.13531134d0,.24705169d0,.38021171d0,.52379159d0,.66577472d0,
     *.79419019d0,.89816102d0,.96884798d0/
      DATA WTLN(1),WTLN(2),WTLN(3),WTLN(4),WTLN(5),WTLN(6),WTLN(7),
     *WTLN(8),WTLN(9),WTLN(10)/.12095474d0,.18636310d0,.19566066d0,
     *.17357723d0,.13569597d0,.093647084d0,.055787938d0,.027159893d0,
     *.0095151992d0,.0016381586d0/
C
C  ETA2 AND WT2 ARE 10TH ORDER GAUSSIAN QUADRATURE
C  VALUES AND WEIGHTS
C
      DATA ETA2(1),ETA2(2),ETA2(3),ETA2(4),ETA2(5),ETA2(6),ETA2(7),
     *ETA2(8),ETA2(9),ETA2(10)/-.97390653d0,-.86506337d0,-.67940957d0,
     *-.43339539d0,-.14887434d0,.14887434d0,.43339539d0,.67940957d0,
     *.86506337d0,.97390653d0/
      DATA WT2(1),WT2(2),WT2(3),WT2(4),WT2(5),WT2(6),WT2(7),WT2(8),
     *WT2(9),WT2(10)/.06667134d0,.14945135d0,.21908636d0,.26926672d0,
     *.29552422d0,.29552422d0,.26926672d0,.21908636d0,.14945135d0,
     *.06667134d0/
      DATA PI/3.14159265d0/
C
C  INITIALIZE MATRICES
C
      DO 10 I=1,NNODES
        DO 10 J=1,NNODES
          G(I,J)=0.0
          H(I,J)=0.0
 10   CONTINUE
C
C  BEGIN LOOPS ON NODE I AND ELEMENT J
C  FOR EACH I ROW, PERFORM INTEGRATIONS OVER EACH ELEMENT J
C  AROUND THE STRUCTURE.  FILL IN ONE ROW AT A TIME
C
      DO 1000 I=1,NNODES
        DO 1000 J=1,NELEMS
C
C  DETERMINE IF THE INTEGRATION SHOULD BE SINGULAR
C  I.E. IS THE NODE I SOMEWHERE ON THE ELEMENT?
C  ISING = 0 MEANS NODE I IS NOT ON ELEMENT J
C  ISING = 1 MEANS NODE I IS FIRST NODE ON ELEMENT J
C  ISING = 2 MEANS NODE I IS SECOND NODE ON ELEMENT J
C  ISING = 3  ETC
C
          ISING=0
          DO 20 L=1,ORDER
            NODES=J*ORDER-(ORDER-L)
            IF(I.EQ.NODES) ISING=L
 20       CONTINUE
          IF(ISING.EQ.0) THEN
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
C  THE INTEGRATION IS NOT SINGULAR
C  PERFORM 12TH ORDER GAUSSIAN QUADRATURE
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
            DO 50 K=1,12
C
C  CALCULATE THE X AND Y COORDINATES OF THE QUADRATURE POINT
C  AND CALCULATE DISTANCE FROM NODE I TO QUAD POINT
C
C  CALCULATE SHAPE FUNCTIONS AT QUAD POINT (ETA(K)) BASED ON GEOMETRY
C  NODES AT -1,0,1
C
              ETAA=-1.
              ETAC=1.
              ETAQ=ETA(K)
              SHAP(1)=(ETAQ-ETAC)/(ETAA-ETAC)
              SHAP(2)=(ETAQ-ETAA)/(ETAC-ETAA)
              X=SHAP(1)*ELEM(J,1)+SHAP(2)*ELEM(J,3)
```

```
                Y=SHAP(1)*ELEM(J,2)+SHAP(2)*ELEM(J,4)
                RI=XI(I)-X
                RJ=YI(I)-Y
                R=SQRT(RI*RI+RJ*RJ)
C
C  CALCULATE PHI* AND Q*
C
                CALL JACOBIAN(J,ETA(K))
                PHISTAR=dlog(1/R)
                D=RI*(NORMI)+RJ*(NORMJ)
                QSTAR=(D/(R*R))
C
C  CALCULATE TERMS OF H AND G MATRICES
C  ROW CORRESPONDS TO NODE I
C  COLUMN CORRESPONDS TO GLOBAL NUMBER OF ELEMENT NODES
C
                CALL SHAPE(ORDER,ETA(K),SHAP)
                DO 35 L=1,ORDER
C
C  JJ IS GLOBAL NUMBER OF ELEMENT NODES
C  FOR EXAMPLE; NODES 1,2,3 ON ELEMENT 1 ARE 1,2,3
C                 NODES 1,2,3 ON ELEMENT 2 ARE 4,5,6
C                 NODES 1,2,3 ON ELEMENT 3 ARE 7,8,9
C
                 JJ=J*ORDER-(ORDER-L)
                 G(I,JJ)=G(I,JJ)+SHAP(L)*PHISTAR*JAC*WT(K)/(2.*PI)
                 H(I,JJ)=H(I,JJ)+SHAP(L)*QSTAR*JAC*WT(K)/(2.*PI)
 35             CONTINUE
 50            CONTINUE
             ELSE
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
C
C  THE INTEGRAL IS SINGULAR IN THE INSIDES OF ELEMENT J
C
C  DETERMINE THE ETA BAR LOCATION OF SINGULARITY
C
               SINGUL=NODE(ISING)
C
C  CALCULATE TERMS OF G MATRIX
C
C
               X2X1=ELEM(J,3)-ELEM(J,1)
               Y2Y1=ELEM(J,4)-ELEM(J,2)
               PLENG=SQRT(X2X1*X2X1+Y2Y1*Y2Y1)
               DO 350 K=1,10
                ET=SINGUL+(1.-SINGUL)*ETALN(K)
                CALL SHAPE(ORDER,ET,SHPLN2)
                CALL JACOBIAN(J,ET)
                JACLN2=JAC
                ET=SINGUL+.5*(1.-SINGUL)*(ETA2(K)+1.)
                CALL SHAPE(ORDER,ET,SHAP)
                CALL JACOBIAN(J,ET)
                JAC4=JAC
                ET=SINGUL-(SINGUL+1.)*ETALN(K)
```

```
                   CALL SHAPE(ORDER,ET,SHPLN)
                   CALL JACOBIAN(J,ET)
                   JACLN=JAC
                   ET=SINGUL-.5*(1.+SINGUL)*(ETA2(K)+1.)
                   CALL SHAPE(ORDER,ET,SHAP3)
                   CALL JACOBIAN(J,ET)
                   JAC3=JAC
                   DO 385 L=1,ORDER
                     JJ=J*ORDER-(ORDER-L)
                     TERM1=SHPLN(L)*JACLN*(SINGUL+1.)*WTLN(K)/(2.*PI)
                     TERM3=SHAP3(L)*JAC3*
     *               dlog(PLENG*ABS(1.+SINGUL)/2.)*(SINGUL+1.)
     *                         *.5*WT2(K)/(2.*PI)
                     TERM2=SHPLN2(L)*JACLN2*(1.-SINGUL)*WTLN(K)/(2.*PI)
                     TERM4=SHAP(L)*JAC4*
     *               dlog(PLENG*ABS(SINGUL-1.)/2.)*(1.-SINGUL)
     *                         *.5*WT2(K)/(2.*PI)
                     G(I,JJ)=G(I,JJ)+TERM1+TERM2-TERM3-TERM4
385                CONTINUE
350              CONTINUE
               END IF
1000  CONTINUE
      RETURN
      END
```

```
C*********************************************************************
C
C  FLOWOUT
C
C  THIS SUBROUTINE OUTPUTS THE POTENTIAL FLUID INFORMATOINS
C
C*********************************************************************
      SUBROUTINE FLOWOUT
      implicit real*8  (a-h,o-z)
      COMMON/GEOMET/NNODES,NELEMS,ORDER,EPS,NORMI,NORMJ,JAC,NODEC,
     *NODPIN1,NODPIN2,NODE(5),ELEM(360,4)
      COMMON/FLOW/H(240,240),G(240,240),Q(240),GHAT(240)
      COMMON/POTEN/VINF,ALPHA,VI,VJ,PINF,RHOINF,GAMMA,MINF2,NODEP
      COMMON/VELOCIT/UVEL(240),VVEL(240),XFORC(480),YFORC(480),
     *XOLD(480),YOLD(480)
      REAL*8 NORMI,NORMJ,JAC,NODE,GAMMA,MINF2
      INTEGER ORDER
C
C
      WRITE(6,1) VINF,ALPHA,VI,VJ,MINF2**.5
 1    FORMAT(//,22X,'***** FLOW INFORMATION *****',/,24X,
     *    '**  ON OUTER BOUNDARY **',/,10X,'Vinfinity = ',F10.4,
     *  5X,'ALPHA = ',F10.4,/,5X,'VI = ',F10.4,10X,'VJ = ',F10.4,
     *  10X,'MINF = ',F10.4,//)
      WRITE(6,2)
 2    FORMAT(1X,'POTENTIAL',3X,'U VEL',5X,'V VEL',4X,'VELOCITY',
     *5X,'CP',6X,'X FORCE',3X,'Y FORCE',//)
      DO 5 I=1,NNODES
        VEL=SQRT(UVEL(I)**2+VVEL(I)**2)
        CP=1.-(VEL/VINF)**2
        WRITE(6,3) GHAT(I),UVEL(I),VVEL(I),VEL,CP,XFORC(I),YFORC(I)
 3      FORMAT(7F10.4)
 5    CONTINUE
C
C  WRITE FORCE ON INSIDES
C
      VELVENT=SQRT(UVEL(NODEP)**2+VVEL(NODEP)**2)
      CPVENT=1.-(VELVENT/VINF)**2
      PVENT=.5*RHOINF*VINF*VINF*CPVENT
      WRITE(6,6) PVENT,NODEP
 6    FORMAT(//,22X,'*****  FORCES ON INNER SURFACE  *****',
     */,10X,'INNER PRESSURE IS = ',F10.4,/,
     *10X,'VENTED NODE IS = ',I3,//)
      WRITE(6,7)
 7    FORMAT(2X,'X FORCE',3X,'Y FORCE',//)
      WRITE(6,8)  (XFORC(I),YFORC(I),I=NNODES+1,2*NNODES)
 8    FORMAT(2F10.4)
      RETURN
      END
```

```
C*****************************************************************
C
C  FLUIDIN
C
C  THIS SUBROUTINE READS IN THE INFORMATION REQUIRED TO
C  CALCULATE THE POTENTIAL FLUID DATA
C
C  INPUT:
C     VINF = FLOW VELOCITY AT INFINITY
C     ALPHA = FLOW ANGLE OF ATTACK
C     PINF = PRESSURE AT INFINITY
C     RHOINF = DENSITY AT INFINITY
C     GAMMA = GAMMA
C     NODEP = NODE AT WHICH TO VENT PRESSURE TO INSIDE
C
C  OUTPUT:
C     VI = X-COMPONENT OF FLOW VELOCITY
C     VJ = Y-COMPONENT OF FLOW VELOCITY
C     MINF2 = MACH AT INFINITY SQUARED
C
C*****************************************************************
      SUBROUTINE FLUIDIN
      implicit real*8  (a-h,o-z)
      COMMON/POTEN/VINF,ALPHA,VI,VJ,PINF,RHOINF,GAMMA,MINF2,NODEP
      real*8 GAMMA,MINF2
C
C  READ IN Vinfinity AND ANGLE OF ATTACK
C  CONVERT Vinf TO X AND Y (I AND J) COMPONENTS
C
      READ(5,*) VINF,ALPHA
 1    FORMAT(2F10.4)
      ALF=ALPHA*0.017453293
      VI=VINF*COS(ALF)
      VJ=VINF*SIN(ALF)
C
C  READ IN Pinfinity, RHOinfinity and GAMMA
C
      READ(5,*) PINF,RHOINF,GAMMA
 2    FORMAT(3F10.4)
      MINF2=(VINF*VINF*RHOINF)/(GAMMA*PINF)
C
C  READ IN NODE AT WHICH TO VENT PRESSURE TO INSIDE
C
      READ(5,*) NODEP
 3    FORMAT(I2)
      RETURN
      END
```

```
C*****************************************************************
C
C  GEOMIN
C
C  THIS SUBROUTINE READS IN THE GEOMETRY INFORMATION
C  OF THE STRUCTURE
C
C  INPUT:
C     ORDER = ORDER OF POTENTIAL AND Q MODELING
C     NELEMS = NUMBER OF ELEMENTS ALONG OUTSIDE OF STRUCTURE
C              TOTAL NUMBER OF ELEMENTS IS 2*NELEM BECAUSE SAME NUMBER
C              ALONG INSIDE OF STRUCTURE
C     EPS = DISTANCE FROM END OF ELEMENT TO THE END NODES
C     NODEC = NODE TO BE REPLACED BY CONTINUITY
C     NODPIN1,NODPIN2 = NODES TO BE PINNED
C
C  OUTPUT:
C     ORDER = NUMBER OF NODES ON EACH ELEMENT
C     NNODES = NUMBER OF NODES
C     NODE(I) = ETA POSITION OF NODE I WITHIN EACH ELEMENT
C     XI,YI = X,Y LOCATION OF NODE I
C
C*****************************************************************
      SUBROUTINE GEOMIN
      implicit real*8  (a-h,o-z)
      COMMON/GEOMET/NNODES,NELEMS,ORDER,EPS,NORMI,NORMJ,JAC,NODEC,
     *NODPIN1,NODPIN2,NODE(5),ELEM(360,4)
      COMMON/POSITION/XI(1080),YI(1080)
      real*8 NORMI,NORMJ,JAC,NODE
      INTEGER ORDER
      DATA PI/3.14159265d0/
C
C  READ IN ORDER OF FIT AND NUMBER OF ELEMENTS ALONG OUTSIDE SURFACE
C
      READ(5,*) IORDER,NELEMS
      ORDER=IORDER+1
      NNODES=ORDER*NELEMS
C
C  CALL SURFACE GEOMETRY GENERATION TO GET LEFT AND RIGHT
C  X,Y LOCATIONS OF OUTSIDE AND INSIDE PANELS
C
      CALL SURF
C
C  DETERMINE LEFT AND RIGHT X,Y OF LINKING PANELS
C  LINK OUTSIDE BOUNDARY WITH INSIDE
C
      DO 10 I=1,NELEMS
        ELEM(2*NELEMS+I,1)=ELEM(I,3)
        ELEM(2*NELEMS+I,2)=ELEM(I,4)
        ELEM(2*NELEMS+I,3)=ELEM(NELEMS+I,3)
        ELEM(2*NELEMS+I,4)=ELEM(NELEMS+I,4)
10    CONTINUE
C
```

```
C  READ IN DISTANCE OF END NODES TO ELEMENT CORNERS
C
      READ(5,*) EPS
      IF(ORDER.EQ.1) EPS=1.
C
C  CALCULATE THE ETA POSITION OF THE NODES WITHIN THE ELEMENT
C  NODE(1) AND NODE(ORDER) ARE EPS FROM +/- 1.  OTHER NODES
C  ARE SPACED EQUALLY WITHIN THE REMAINING SPACE
C
      IF(ORDER.EQ.1) THEN
        NODE(1)=0.
      ELSE
        SPACE=(2.-2.*EPS)/(ORDER-1.)
        NODE(1)=-(1.-EPS)
        NODE(ORDER)=-NODE(1)
        IF(ORDER.GT.2) THEN
          DO 20 I=2,ORDER-1
            NODE(I)=NODE(I-1)+SPACE
 20       CONTINUE
        END IF
      END IF
C
C  CALCULATE THE X,Y POSITION OF EACH NODE
C
      DO 40 I=1,NELEMS*3
        DO 40 L=1,ORDER
          ETAA=-1.
          ETAC=1.
          ETAQ=NODE(L)
          SHAP1=(ETAQ-ETAC)/(ETAA-ETAC)
          SHAP2=(ETAQ-ETAA)/(ETAC-ETAA)
          XI(I*ORDER-(ORDER-L))=SHAP1*ELEM(I,1)+SHAP2*ELEM(I,3)
          YI(I*ORDER-(ORDER-L))=SHAP1*ELEM(I,2)+SHAP2*ELEM(I,4)
 40   CONTINUE
C
C  READ IN THE NODE OF EACH ELEMENT TO BE REPLACED BY THE
C  CONTINUITY EQUATION
C
      READ(5,*) NODEC
C
C  READ IN NODES TO BE PINNED
C
      READ(5,*) NODPIN1,NODPIN2
      RETURN
      END
```

```
C**********************************************************************
C
C  GOUTPUT
C
C  THIS SUBROUTINE OUTPUTS THE ELEMENT GEOMETRY COORDINATES
C
C**********************************************************************
      SUBROUTINE GOUTPUT(ITER)
      implicit real*8  (a-h,o-z)
      COMMON/GEOMET/NNODES,NELEMS,ORDER,EPS,NORMI,NORMJ,JAC,NODEC,
     *NODPIN1,NODPIN2,NODE(5),ELEM(360,4)
      COMMON/POSITION/XI(1080),YI(1080)
      real*8 NORMI,NORMJ,JAC,NODE
      INTEGER ORDER
C
C
      WRITE(6,10) NELEMS,3*NELEMS,ORDER,NNODES,3*NNODES,EPS,ITER-1
 10   FORMAT(//,22X,'*****GEOMETRY INFORMATION*****',//,
     *10X,'NUMBER OF ELEMENTS ON OUTSIDE = ',I3,
     *10X,'TOTAL NUMBER OF ELEMENTS = ',I3,
     */,10X,'ORDER OF MODELING = ',I3,
     */,10X,'NUMBER OF NODES ALONG OUTSIDE = ',I3,
     *10X,'TOTAL NUMBER OF NODES = ',I3,
     */,10X,'DISTANCE FROM END NODES',
     *' TO CORNER = ',F10.4,/,10X,' ITERATION NUMBER = ',I2,/)
      WRITE(6,20)
 20   FORMAT(//,5X,'ELEMENT COORDINATES',/,6X,'ELEMENT',7X,'X-LEFT',
     *7X,'Y-LEFT',6X,'X-RIGHT',6X,'Y-RIGHT',//)
      DO 30 I=1,NELEMS*2
        WRITE(6,25) I,ELEM(I,1),ELEM(I,2),ELEM(I,3),ELEM(I,4)
 25     FORMAT(9X,I3,5X,F10.4,3(3X,F10.4))
 30   CONTINUE
      RETURN
      END
```

```
C*****************************************************************
C
C  GSHATT
C
C  THIS SUBROUTINE MOVES THE CONNECTING ELEMENT TERMS OF THE GS
C  MATRIX OVER TO THE HS MATRIX AND THEN CALCULATES GSHAT
C
C*****************************************************************
      SUBROUTINE GSHATT
      implicit real*8  (a-h,o-z)
      COMMON/GEOMET/NNODES,NELEMS,ORDER,EPS,NORMI,NORMJ,JAC,NODEC,
     *NODPIN1,NODPIN2,NODE(5),ELEM(360,4)
      COMMON/VELOCIT/UVEL(240),VVEL(240),XFORC(480),YFORC(480),
     *XOLD(480),YOLD(480)
      COMMON/STRUCT/GE,XNU,HS(1920,1920),GS(1920,1440),GSHAT(1920)
      INTEGER ORDER
      REAL*8 NODE,NORMI,NORMJ,JAC
C
C  MOVE THE CONNECTING ELEMENT TERMS OF THE GS MATRIX OVER TO
C  THE HS MATRIX
C
      DO 10  I=1,8*NNODES
        DO 10 J=1,NNODES
          HS(I,3*NNODES+J)=-GS(I,2*NNODES+J)
          HS(I,7*NNODES+J)=-GS(I,5*NNODES+J)
10    CONTINUE
C
C  CALCULATE THE GSHAT MATRIX
C  REMEMBER THE COLUMNS OF GS CORRESPONDING TO THE CONNECTING
C  ELEMENTS DO NOT COUNT
C
C  SUBTRACT OLD TRACTIONS (XOLD,YOLD) FROM NEW TRACTIONS (XFORC,YFORC)
C  THIS PRE-STRESSES THE STRUCTURE TO ACCOUNT FROM THE DEFORMATION AND
C  STRESS DUE TO THE LAST PASS
C
      DO 20 I=1,8*NNODES
        GSHAT(I)=0.0
        DO 20 J=1,2*NNODES
          GSHAT(I)=GSHAT(I)+GS(I,J)*(XFORC(J)-XOLD(J))+
     *              GS(I,3*NNODES+J)*(YFORC(J)-YOLD(J))
20    CONTINUE
      RETURN
      END
```

```
C***********************************************************************
C
C  HDIAG
C
C  THIS SUBROUTINE CALCULATES THE DIAGONAL TERMS OF THE H MATRIX
C
C***********************************************************************
      SUBROUTINE HDIAG
      implicit real*8  (a-h,o-z)
      COMMON/GEOMET/NNODES,NELEMS,ORDER,EPS,NORMI,NORMJ,JAC,NODEC,
     *NODPIN1,NODPIN2,NODE(5),ELEM(360,4)
      COMMON/FLOW/H(240,240),G(240,240),Q(240),GHAT(240)
      integer order
      real*8 JAC,NORMI,NORMJ,NODE
C
C  EXTERNAL FLOW (UNBOUNDED INFINITE DOMAIN)
C  HII=-SUM(HIJ)+1
C
        DO 10 I=1,NNODES
          SUM=0.0
          DO 3 J=1,NNODES
            IF(I.NE.J) SUM = SUM+H(I,J)
 3        CONTINUE
          H(I,I)=(-SUM+1.)
10      CONTINUE
      RETURN
      END
```

```
C*******************************************************************
C
C  HGQOUT
C
C  THIS SUBROUTINE OUTPUTS THE POTENTIAL FLOW MATRICES
C   H, G, Q, GHAT
C
C*******************************************************************
      SUBROUTINE HGQOUT
      implicit real*8  (a-h,o-z)
      COMMON/GEOMET/NNODES,NELEMS,ORDER,EPS,NORMI,NORMJ,JAC,NODEC,
     *NODPIN1,NODPIN2,NODE(5),ELEM(360,4)
      COMMON/FLOW/H(240,240),G(240,240),Q(240),GHAT(240)
      real*8 NORMI,NORMJ,JAC,NODE
      INTEGER ORDER
C
C
        WRITE(6,1)
 1      FORMAT(//,22X,'*****POTENTIAL FLOW MATRICES*****',/,
     *             22X,'** FOR OUTER SURFACE BOUNDARY **',//)
        WRITE(6,2)
 2      FORMAT(5X,'H MATRIX',/)
        DO 10 I=1,NNODES
            WRITE(6,3) (H(I,J),J=1,NNODES)
 3          FORMAT(/13(8(F10.4,5X),/))
 10     CONTINUE
        WRITE(6,11)
 11     FORMAT(//,5X,'G MATRIX',/)
        DO 20 I=1,NNODES
            WRITE(6,12) (G(I,J),J=1,NNODES)
 12         FORMAT(/13(8(F10.4,5X),/))
 20     CONTINUE
        WRITE(6,21)
 21     FORMAT(//,5X,'Q MATRIX',/)
          WRITE(6,22) (Q(I),I=1,NNODES)
 22       FORMAT(13(8(F10.4,5X),/))
        WRITE(6,31)
 31     FORMAT(//,5X,'GHAT MATRIX',/)
          WRITE(6,32) (GHAT(I),I=1,NNODES)
 32       FORMAT(13(8(F10.4,5X),/))
      RETURN
      END
```

```
C**********************************************************************
C
C  HSDIAG
C
C  THIS SUBROUTINE CALCULATES THE DIAGONAL TERM OF THE HS MATRIX
C
C  HII = -SUM(HIJ)
C
C  FOR THE HXX AND HYY TERMS ONLY BECAUSE CXY=CYX=0. AND THEREFORE
C  THE DIAGONAL TERMS OF HXY AND HYX ARE EQUAL TO THEMSELVES
C
C**********************************************************************
      SUBROUTINE HSDIAG
      implicit real*8  (a-h,o-z)
      COMMON/GEOMET/NNODES,NELEMS,ORDER,EPS,NORMI,NORMJ,JAC,NODEC,
     *NODPIN1,NODPIN2,NODE(5),ELEM(360,4)
      COMMON/STRUCT/GE,XNU,HS(1920,1920),GS(1920,1440),GSHAT(1920)
      real*8 NORMI,NORMJ,JAC,NODE
      INTEGER ORDER
C
C  SUM FOR THE OUTER, INNER AND CONNECTING ROWS
C
      DO 10 I=1,3*NNODES
        SUMXX=0.0
        SUMYY=0.0
        DO 5 J=1,3*NNODES
          IF(I.NE.J) THEN
            SUMXX=SUMXX+HS(I,J)
            SUMYY=SUMYY+HS(4*NNODES+I,4*NNODES+J)
          END IF
 5      CONTINUE
        HS(I,I)=-SUMXX
        HS(4*NNODES+I,4*NNODES+I)=-SUMYY
10    CONTINUE
C
C  CONTINUE TO SUM FOR THE ADDITIONAL SET OF CONNECTING ROWS
C
      DO 20 I=3*NNODES+1,4*NNODES
        II=I-NNODES
        SUMXX=0.0
        SUMYY=0.0
        DO 15 J=1,3*NNODES
          IF(II.NE.J) THEN
            SUMXX=SUMXX+HS(I,J)
            SUMYY=SUMYY+HS(4*NNODES+I,4*NNODES+J)
          END IF
15      CONTINUE
        HS(I,II)=-SUMXX
        HS(4*NNODES+I,4*NNODES+II)=-SUMYY
20    CONTINUE
      RETURN
      END
```

```
C***********************************************************************
C
C  JACOBIAN
C
C  THIS SUBROUTINE CALCULATES THE JACOBIAN AND OUTWARD NORMAL
C  FOR THE POINT UNDER CONSIDERATION
C
C  INPUT:
C  J = ELEMENT ON WHICH POINT IS LOCATED
C  ETA = POINT WHERE JACOBIAN AND NORMAL IS CALCULATED
C
C  OUTPUT:
C  JAC = JACOBIAN
C  NORMI = X COMPONENT OF NORMAL
C  NORMJ = Y COMPONENT OF NORMAL
C
C***********************************************************************
      SUBROUTINE JACOBIAN(J,ETA)
      implicit real*8  (a-h,o-z)
      COMMON/GEOMET/NNODES,NELEMS,ORDER,EPS,NORMI,NORMJ,JAC,NODEC,
     *NODPIN1,NODPIN2,NODE(5),ELEM(360,4)
      real*8 NORMI,NORMJ,JAC,NODE
      INTEGER ORDER
C
C  ASSUME FLAT PANEL
C
      DN1=-.5
      DN3=.5
      DXDN=DN1*ELEM(J,1)+DN3*ELEM(J,3)
      DYDN=DN1*ELEM(J,2)+DN3*ELEM(J,4)
      JAC=SQRT(DXDN*DXDN+DYDN*DYDN)
      NORMI=-DYDN/JAC
      NORMJ=DXDN/JAC
      RETURN
      END
```

```
C*******************************************************************
C
C  PRESSURE
C
C  THIS SUBROUTINE CALCULATES THE COEFFICIENT OF PRESSURE (CP)
C  AND THE EXTERNAL FORCES (XFORC,YFORC) ON EACH NODE
C
C*******************************************************************
      SUBROUTINE PRESSURE
      implicit real*8  (a-h,o-z)
      COMMON/GEOMET/NNODES,NELEMS,ORDER,EPS,NORMI,NORMJ,JAC,NODEC,
     *NODPIN1,NODPIN2,NODE(5),ELEM(360,4)
      COMMON/POTEN/VINF,ALPHA,VI,VJ,PINF,RHOINF,GAMMA,MINF2,NODEP
      COMMON/VELOCIT/UVEL(240),VVEL(240),XFORC(480),YFORC(480),
     *XOLD(480),YOLD(480)
      REAL*8  JAC,NORMI,NORMJ,GAMMA,MINF2,NODE
      INTEGER ORDER
C
C  STORE THE OLD VALUES OF TRACTIONS
C
      DO 10 I=1,2*NNODES
        XOLD(I)=XFORC(I)
        YOLD(I)=YFORC(I)
 10   CONTINUE
C
C  INITIALIZE XFORC,YFORC
C
      DO 15 I=1,2*NNODES
        XFORC(I)=0.0
        YFORC(I)=0.0
 15   CONTINUE
C
C  CALCULATE PRESSURE AND X,Y COMPONENTS OF PRESSURE FOR OUTER SURFACE
C  XFORCE = NI*P,  YFORC = NJ*P
C
      DO 20 J=1,NELEMS
        DO 20 K=1,ORDER
          JJJ=J*ORDER-(ORDER-K)
          VEL=SQRT(UVEL(JJJ)*UVEL(JJJ)+VVEL(JJJ)*VVEL(JJJ))
          CP=1.-(VEL/VINF)**2
          P=0.5*RHOINF*VINF*VINF*CP
          CALL JACOBIAN(J,NODE(K))
          XFORC(JJJ)=XFORC(JJJ)+P*(NORMI)
          YFORC(JJJ)=YFORC(JJJ)+P*(NORMJ)
 20   CONTINUE
C
C  CALCULATE PRESSURES FOR INSIDE BOUNDARY WHERE INTERNAL PRESSURE
C  EQUALS PRESSURE AT VENTED NODE ON OUTER SURFACE
C
      VELVENT=SQRT(UVEL(NODEP)*UVEL(NODEP)+VVEL(NODEP)*VVEL(NODEP))
      CPVENT=1.-(VELVENT/VINF)**2
      PIN=.5*RHOINF*VINF*VINF*CPVENT
      DO 30 J=NELEMS+1,2*NELEMS
        DO 30 K=1,ORDER
          JJJ=J*ORDER-(ORDER-K)
          CALL JACOBIAN(J,NODE(K))
          XFORC(JJJ)=XFORC(JJJ)+PIN*(-NORMI)
          YFORC(JJJ)=YFORC(JJJ)+PIN*(-NORMJ)
   30   CONTINUE
        RETURN
        END
```

```
C***************************************************************************
C
C  QFLOW
C
C  THIS SUBROUTINE CALCULATES THE BOUNDARY CONDITION VECTOR,Q
C  USING Q = -n . Vinf
C
C  NOTE THAT SINCE THE OUTWARD NORMALS TO THE FLUID FLOW POINT INTO THE
C  BODY THE NORMAL CALCULATED IN JACOBIAN SUBROUTINE ALREADY HAS THE
C  PROPER SIGN (-N)
C
C**************************************************************************
      SUBROUTINE QFLOW
      implicit real*8  (a-h,o-z)
      COMMON/GEOMET/NNODES,NELEMS,ORDER,EPS,NORMI,NORMJ,JAC,NODEC,
     *NODPIN1,NODPIN2,NODE(5),ELEM(360,4)
      COMMON/POTEN/VINF,ALPHA,VI,VJ,PINF,RHOINF,GAMMA,MINF2,NODEP
      COMMON/FLOW/H(240,240),G(240,240),Q(240),GHAT(240)
      INTEGER ORDER
      real*8 JAC,NORMI,NORMJ,GAMMA,MINF2,NODE
C
C  EXTERNAL FLOW PROBLEM
C  Q=NORMAL*VINF
C
        DO 5 I=1,NNODES
          Q(I)=0.0
5       CONTINUE
        DO 10 J=1,NELEMS
          DO 10 L=1,ORDER
            JJ=J*ORDER-(ORDER-L)
            CALL JACOBIAN(J,NODE(L))
            Q(JJ)=Q(JJ)+(NORMI*VI+NORMJ*VJ)
 10     CONTINUE
      RETURN
      END
```

```
C*******************************************************************
C
C  SFORM
C
C  THIS SUBROUTINE CALCULATES THE H AND G MATRICES FOR THE
C  ELASTIC STRUCTURE
C
C*******************************************************************
      SUBROUTINE SFORM
      implicit real*8  (a-h,o-z)
      COMMON/GEOMET/NNODES,NELEMS,ORDER,EPS,NORMI,NORMJ,JAC,NODEC,
     *NODPIN1,NODPIN2,NODE(5),ELEM(360,4)
      COMMON/STRUCT/GE,XNU,HS(1920,1920),GS(1920,1440),GSHAT(1920)
      COMMON/POSITION/XI(1080),YI(1080)
      COMMON/VELOCIT/UVEL(240),VVEL(240),XFORC(480),YFORC(480),
     *XOLD(480),YOLD(480)
      REAL*8 ETA(12),WT(12),ETALN(10),WTLN(10),ETAS(8),WTS(8),
     *ETA2(10),WT2(10)
      REAL*8 SHAP(5),SHAP2(5),SHAPLN(5),SHAPLN2(5),SHAP3(5),
     *   SHAP4(5)
      REAL*8 NODE,JAC,JAC1,JAC2,JAC3,JAC4,JACLN,JACLN2,NORMI,NORMJ,
     *NORMI1,NORMJ1,NORMI2,NORMJ2
      INTEGER ORDER
C
C  ETA AND WT ARE 12TH ORDER GAUSSIAN QUADRATRUE VALUES AND WEIGHTS
C
      DATA ETA(1),ETA(2),ETA(3),ETA(4),ETA(5),ETA(6),ETA(7),
     *ETA(8),ETA(9),ETA(10),ETA(11),ETA(12)/-.98156063d0,-.90411726d0,
     *-.76990267d0,-.58731795d0,-.36783149d0,-.12523341d0,.12523341d0,
     *.36783149d0,.58731795d0,.76990267d0,.90411726d0,.98156063d0/
      DATA WT(1),WT(2),WT(3),WT(4),WT(5),WT(6),WT(7),WT(8),WT(9),
     *WT(10),WT(11),WT(12)/.0471753d0,.10693933d0,.16007833d0,
     *.20316743d0,.23349254d0,.24914705d0,.24914705d0,.23349254d0,
     *.20316743d0,.16007833d0,.10693933d0,.04717534d0/
C
C  ETALN AND WTLN ARE 10TH ORDER LOGARITHMIC GAUSSIAN QUADRATURE
C  VALUES AND WEIGHTS
C
      DATA ETALN(1),ETALN(2),ETALN(3),ETALN(4),ETALN(5),ETALN(6),
     *ETALN(7),ETALN(8),ETALN(9),ETALN(10)/.00904259d0,.05397105d0,
     *.13531134d0,.24705169d0,.38021171d0,.52379159d0,.66577472d0,
     *.79419019d0,.89816102d0,.96884798d0/
      DATA WTLN(1),WTLN(2),WTLN(3),WTLN(4),WTLN(5),WTLN(6),WTLN(7),
     *WTLN(8),WTLN(9),WTLN(10)/.12095474d0,.18636310d0,.19566066d0,
     *.17357723d0,.13569597d0,.09364708d0,.055787938d0,.02715989d0,
     *.0095151992d0,.0016381586d0/
C
C  ETAS AND WTS ARE 8TH ORDER GAUSSIAN VALUES AND WEIGHTS FOR
C  1/R SINGULAR INTEGRALS
C
      DATA ETAS(1),ETAS(2),ETAS(3),ETAS(4),ETAS(5),ETAS(6),ETAS(7),
     *ETAS(8)/-.00324250d0,.05349077d0,.17782733d0,.35071788d0,
     *.5458195sd0,.73342708d0,.88498305d0,.97745438d0/
```

```
      DATA WTS(1),WTS(2),WTS(3),WTS(4),WTS(5),WTS(6),WTS(7),WTS(8)
     */-3.93063681d0,1.73740631d0,.85763454d0,.53836077d0,
     *.35975759d0,.23726769d0,.14146228d0,.05874761d0/
C
C  ETA2 AND WT2 ARE 10TH ORDER GAUSSIAN QUADRATURES
C
      DATA ETA2(1),ETA2(2),ETA2(3),ETA2(4),ETA2(5),ETA2(6),ETA2(7),
     *ETA2(8),ETA2(9),ETA2(10)/-.97390653d0,-.86506337d0,-.67940957d0,
     *-.43339539d0,-.14887434d0,.14887434d0,.43339539d0,.67940957d0,
     *.86506337d0,.97390653d0/
      DATA WT2(1),WT2(2),WT2(3),WT2(4),WT2(5),WT2(6),WT2(7),WT2(8),
     *WT2(9),WT2(10)/.06667134d0,.14945135d0,.21908636d0,.26926672d0,
     *.29552423d0,.29552423d0,.26926672d0,.21908636d0,.14945135d0,
     *.06667134d0/
      DATA PI/3.14159265d0/
C
C  INITIALIZE MATRICES
C
      DO 15 I=1,8*NNODES
        DO 10 J=1,6*NNODES
          GS(I,J)=0.0
 10     CONTINUE
        DO 15 J=1,8*NNODES
          HS(I,J)=0.0
 15   CONTINUE
C
C  BEGIN LOOPS ON NODE I AND ELEMENT J
C
      DO 1000 III=1,4
        DO 1000 II=1,NELEMS
          DO 1000 K=1,ORDER
            I=(III-1)*NNODES+II*ORDER-(ORDER-K)
            IF(III.EQ.4) THEN
              INODE=2*NNODES+II*ORDER-(ORDER-K)
            ELSE
              INODE=I
            END IF
            DO 1000 L=1,4
              IF(III.EQ.4) THEN
                J=NELEMS*(L-1)+II+1
                IF(L.EQ.4.AND.II.NE.NELEMS) J=2*NELEMS+II
                IF(II.EQ.NELEMS) J=NELEMS*(L-1)+1
                IF(II.EQ.NELEMS.AND.L.EQ.4) J=3*NELEMS
              ELSE
                J=NELEMS*(L-1)+II
                IF(L.EQ.4) J=2*NELEMS+(II-1)
                IF(L.EQ.4.AND.II.EQ.1) J=3*NELEMS
              END IF
C
C  DETERMINE IF THE INTEGRATION SHOULD BE SINGULAR
C
C  I.E. IS THE NODE I SOMEWHERE ON THE ELEMENT J
C  ISING = 0 MEANS THE NODE I IS NOT ON ELEMENT J
C  ISING = 1 MEANS THE NODE I IS THE FIRST NODE ON ELEMENT J
```

```
C  ISING = 2 MEANS THE NODE I IS THE SECOND NODE ON ELEMENT J
C  ISING = 3 ETC
C
              ISING=0
              DO 20 LL=1,ORDER
                NODES=J*ORDER-(ORDER-LL)
                IF(INODE.EQ.NODES) ISING=LL
 20           CONTINUE
              IF(ISING.EQ.0) THEN
C ********************************
C  THE INTEGRATION IS NOT SINGULAR
C  PERFORM 12TH ORDER GAUSSIAN QUADRATURE
C ********************************
                DO 50 KK=1,12
C
C  CALCULATE THE X AND Y COORDINATES OF THE QUADRATURE POINT
C  AND CALCULATE THE DISTANCE FROM NODE I TO QUAD POINT
C
                  ETAA=-1.
                  ETAC=1.
                  ETAQ=ETA(KK)
                  SHAP(1)=(ETAQ-ETAC)/(ETAA-ETAC)
                  SHAP(2)=(ETAQ-ETAA)/(ETAC-ETAA)
                  X=SHAP(1)*ELEM(J,1)+SHAP(2)*ELEM(J,3)
                  Y=SHAP(1)*ELEM(J,2)+SHAP(2)*ELEM(J,4)
                  RI=X-XI(INODE)
                  RJ=Y-YI(INODE)
                  R=SQRT(RI*RI+RJ*RJ)
C
C  CALCULATE U* AND P*
C
                  CALL JACOBIAN(J,ETA(KK))
                  IF(L.EQ.1.OR.L.EQ.3) THEN
                    NORMI=-NORMI
                    NORMJ=-NORMJ
                  END IF
                  TERM1=(-1.)/(8.*PI*GE*(1.-XNU))
C
C  NOTE THAT ON THE LEFT CONNECTING ELEMENT (L=4)  Qij=-Qji
C  SO THAT THE GS TERM SHOULD BE NEGATIVE
C
                  IF(L.EQ.4) TERM1=-TERM1
                  USTXX=TERM1*((3.-4.*XNU)*dlog(1./R)
     *                +(RI/R)**2)
                  USTXY=TERM1*((RI*RJ)/(R*R))
                  USTYX=USTXY
                  USTYY=TERM1*((3.-4.*XNU)*dlog(1./R)
     *                +(RJ/R)**2)
                  DRDN=(NORMI*RI+NORMJ*RJ)/R
                  TERM2=(-1.)/(4.*PI*(1.-XNU)*R)
                  PSTXX=TERM2*(DRDN*((1.-2.*XNU)+
     *                2.*(RI/R)**2))
                  PSTXY=TERM2*(DRDN*2.*(RI*RJ)/(R*R)
     *                -(1.-2.*XNU)*((RI*NORMJ-RJ*NORMI)/R))
```

```
                 PSTYX=TERM2*(DRDN*2.*(R1*RJ)/(R*R)
     *               -(1.-2.*XNU)*((RJ*NORMI-RI*NORMJ)/R))
                 PSTYY=TERM2*(DRDN*((1.-2.*XNU)+
     *               2.*(RJ/R)**2))
C
C  CALCULATE THE TERMS OF H AND G MATRICES
C  ROW I CORRESPONDS TO THE X DEFORMATIONS OF NODE I
C  ROW NNODES+I CORRESPONDS TO THE Y DEFORMATIONS OF NODE I
C  COLUMN JJ CORRESPONDS TO THE X TERMS
C  COLUMN NNODES + JJ CORRESPONDS TO THE Y TERMS
C
C  FOR EXAMPLE,
C
C  ---                                                     ---
C  ¶ PXX*1  PXX*2  . . PXX*ORDER  PXY*1  PXY*2 . . PXY*ORDER ¶
C  ¶ PYX*1  PYX*2  . . PYX*ORDER  PYY*1  PYY*2 . . PYY*ORDER ¶
C  --                                                      ---
C
                 CALL SHAPE(ORDER,ETA(KK),SHAP)
                 DO 35 LL=1,ORDER
                   JJ=J*ORDER-(ORDER-LL)
                   HS(I,JJ)=HS(I,JJ)+PSTXX*SHAP(LL)*JAC*WT(KK)
                   HS(I,4*NNODES+JJ)=HS(I,4*NNODES+JJ)
     *                   +PSTXY*SHAP(LL)*JAC*WT(KK)
                   HS(4*NNODES+I,JJ)=HS(4*NNODES+I,JJ)
     *                   +PSTYX*SHAP(LL)*JAC*WT(KK)
                   HS(4*NNODES+I,4*NNODES+JJ)=
     *                   HS(4*NNODES+I,4*NNODES+JJ)
     *                   +PSTYY*SHAP(LL)*JAC*WT(KK)
                   GS(I,JJ)=GS(I,JJ)+USTXX*SHAP(LL)*JAC*WT(KK)
                   GS(I,3*NNODES+JJ)=GS(I,3*NNODES+JJ)
     *                   +USTXY*SHAP(LL)*JAC*WT(KK)
                   GS(4*NNODES+I,JJ)=GS(4*NNODES+I,JJ)
     *                   +USTYX*SHAP(LL)*JAC*WT(KK)
                   GS(4*NNODES+I,3*NNODES+JJ)=
     *                   GS(4*NNODES+I,3*NNODES+JJ)
     *                   +USTYY*SHAP(LL)*JAC*WT(KK)
 35              CONTINUE
 50            CONTINUE
             ELSE
C ***********************************
C  INTEGRAL IS SINGULAR INSIDE THE ELEMENT
C
C  CALCUALTE HS AND GS MATRICES SEPARATELY BECAUSE
C  GS IS LOG(1/R) SINGULAR AND HS IS (1/R) SINGULAR
C
C  CALCULATE GS FIRST
C
C ***********************************
               SINGUL=NODE(ISING)
               X2X1=ELEM(J,3)-ELEM(J,1)
               Y2Y1=ELEM(J,4)-ELEM(J,2)
               PLENG=SQRT(X2X1*X2X1+Y2Y1*Y2Y1)
               DO 350 KK=1,10
```

```
                    CALL SHAPE(ORDER,ETA2(KK),SHAP)
                    CALL JACOBIAN(J,ETA2(KK))
                    JAC2=JAC
                    ET=SINGUL-(SINGUL+1.)*ETALN(KK)
                    CALL SHAPE(ORDER,ET,SHAPLN)
                    CALL JACOBIAN(J,ET)
                    JACLN=JAC
                    ET=SINGUL+(1.-SINGUL)*ETALN(KK)
                    CALL SHAPE(ORDER,ET,SHAPLN2)
                    CALL JACOBIAN(J,ET)
                    JACLN2=JAC
                    ET=SINGUL-(1.+SINGUL)*.5*(ETA2(KK)+1.)
                    CALL SHAPE(ORDER,ET,SHAP3)
                    CALL JACOBIAN(J,ET)
                    JAC3=JAC
                    ET=SINGUL+(1.-SINGUL)*.5*(ETA2(KK)+1.)
                    CALL SHAPE(ORDER,ET,SHAP4)
                    CALL JACOBIAN(J,ET)
                    JAC4=JAC
                    DO 325 LL=1,ORDER
                      JJ=J*ORDER-(ORDER-LL)
                      TERM=(-1.)/(8.*PI*GE*(1.-XNU))
                      IF(L.EQ.4) TERM=-TERM
                      TERMX=(X2X1/PLENG)**2*TERM
                      TERMY=(Y2Y1/PLENG)**2*TERM
                      TERM3=(3.-4.*XNU)*dlog(ABS(PLENG*(1.+SINGUL)/2.))
     *                      *TERM
                      TERM4=(3.-4.*XNU)*dlog(ABS(PLENG*(SINGUL-1.)/2.))
     *                      *TERM
                      TERM1=(3.-4.*XNU)*TERM
                      TERM2=(3.-4.*XNU)*TERM
                      GS(I,JJ)=GS(I,JJ)+
     *                  SHAP(LL)*TERMX*JAC2*WT2(KK)-
     *                  SHAP3(LL)*TERM3*JAC3*(1.+SINGUL)*.5*WT2(KK)-
     *                  SHAP4(LL)*TERM4*JAC4*(1.-SINGUL)*.5*WT2(KK)+
     *                  SHAPLN(LL)*TERM1*JACLN*(1.+SINGUL)*WTLN(KK)+
     *                  SHAPLN2(LL)*TERM2*JACLN2*(1.-SINGUL)*WTLN(KK)
                      GS(I,3*NNODES+JJ)=GS(I,3*NNODES+JJ)+
     *                  SHAP(LL)*X2X1*Y2Y1*JAC2*WT2(KK)*TERM/PLENG**2
                      GS(4*NNODES+I,JJ)=GS(4*NNODES+I,JJ)+
     *                  SHAP(LL)*X2X1*Y2Y1*JAC2*WT2(KK)*TERM/PLENG**2
                      GS(4*NNODES+I,3*NNODES+JJ)=
     *                  GS(4*NNODES+I,3*NNODES+JJ)+
     *                  SHAP(LL)*TERMY*JAC2*WT2(KK)-
     *                  SHAP3(LL)*TERM3*JAC3*(1.+SINGUL)*.5*WT2(KK)-
     *                  SHAP4(LL)*TERM4*JAC4*(1.-SINGUL)*.5*WT2(KK)+
     *                  SHAPLN(LL)*TERM1*JACLN*(1.+SINGUL)*WTLN(KK)+
     *                  SHAPLN2(LL)*TERM2*JACLN2*(1.-SINGUL)*WTLN(KK)
 325                CONTINUE
 350              CONTINUE
C
C  CALCULATE THE HS MATRIX
C  FOR SINGULAR PANELS PXX=PYY=0.
C
```

```
                DO 450 KK=1,8
                  TERM=(1.-2.*XNU)/(4.*PI*(1.-XNU))
                  ET=(-1.-SINGUL)*ETAS(KK)+SINGUL
                  CALL JACOBIAN(J,ET)
                  JAC1=JAC
                  NORMI1=NORMI
                  NORMJ1=NORMJ
                  IF(L.EQ.1.OR.L.EQ.3) THEN
                    NORMI1=-NORMI1
                    NORMJ1=-NORMJ1
                  END IF
                  PXY1=TERM*((X2X1/PLENG)*NORMJ1-(Y2Y1/PLENG)*NORMI1)
                  PYX1=TERM*((Y2Y1/PLENG)*NORMI1-(X2X1/PLENG)*NORMJ1)
                  CALL SHAPE(ORDER,ET,SHAP)
                  ET=(1.-SINGUL)*ETAS(KK)+SINGUL
                  CALL JACOBIAN(J,ET)
                  JAC2=JAC
                  NORMI2=NORMI
                  NORMJ2=NORMJ
                  IF(L.EQ.1.OR.L.EQ.3) THEN
                    NORMI2=-NORMI2
                    NORMJ2=-NORMJ2
                  END IF
                  PXY2=TERM*((X2X1/PLENG)*NORMJ2-(Y2Y1/PLENG)*NORMI2)
                  PYX2=TERM*((Y2Y1/PLENG)*NORMI2-(X2Y1/PLENG)*NORMJ2)
                  CALL SHAPE(ORDER,ET,SHAP2)
                  DO 425 LL=1,ORDER
                    JJ=J*ORDER-(ORDER-LL)
                    HS(I,4*NNODES+JJ)=HS(I,4*NNODES+JJ)-
     *                SHAP(LL)*PXY1*JAC1*(2./PLENG)*WTS(KK)+
     *                SHAP2(LL)*PXY2*JAC2*(2./PLENG)*WTS(KK)
                    HS(4*NNODES+I,JJ)=HS(4*NNODES+I,JJ)-
     *                SHAP(LL)*PYX1*JAC1*(2./PLENG)*WTS(KK)+
     *                SHAP2(LL)*PYX2*JAC2*(2./PLENG)*WTS(KK)
 425              CONTINUE
 450            CONTINUE
C
C  CALCULATE F(SINGULARITY) LOG TERMS AND ADD TO THE HS TERMS
C
                TERM=(1.-2.*XNU)/(4.*PI*(1.-XNU))
                CALL SHAPE(ORDER,SINGUL,SHAP3)
                CALL JACOBIAN(J,SINGUL)
                IF(L.EQ.1.OR.L.EQ.3) THEN
                  NORMI=-NORMI
                  NORMJ=-NORMJ
                END IF
                PXY=TERM*((X2X1/PLENG)*NORMJ-(Y2Y1/PLENG)*NORMI)
                PYX=TERM*((Y2Y1/PLENG)*NORMI-(X2X1/PLENG)*NORMJ)
                DO 475 LL=1,ORDER
                  JJ=J*ORDER-(ORDER-LL)
                  HS(I,4*NNODES+JJ)=HS(I,4*NNODES+JJ)-
     *              SHAP3(LL)*PXY*JAC*(2./PLENG)*dlog(ABS(-1.-SINGUL))+
     *              SHAP3(LL)*PXY*JAC*(2./PLENG)*dlog(ABS(1.-SINGUL))
                  HS(4*NNODES+I,JJ)=HS(4*NNODES+I,JJ)-
     *              SHAP3(LL)*PYX*JAC*(2./PLENG)*dlog(ABS(-1.-SINGUL))+
     *              SHAP3(LL)*PYX*JAC*(2./PLENG)*dlog(ABS(1.-SINGUL))
475           CONTINUE
            END IF
1000 CONTINUE
     RETURN
     END
```

```
C**********************************************************************
C
C  SHAPE
C
C  THIS SUBROUTINE CALCULATES THE SHAPE FUNCTIONS
C  USING LAGRANGIAN POLYNOMIAL SHAPE FUNCTIONS
C
C  INPUT:
C  SORDR = ORDER OF SHAPE FUNCTION
C  TERM = LOCATION WHERE SHAPE FUNCTION IS TO BE CALCULATED
C
C  OUTPUT:
C  SH = SHAPE FUNCTION ARRAY
C
C**********************************************************************
      SUBROUTINE SHAPE(SORDR,TERM,SH)
      implicit real*8  (a-h,o-z)
      COMMON/GEOMET/NNODES,NELEMS,ORDER,EPS,NORMI,NORMJ,JAC,NODEC,
     *NODPIN1,NODPIN2,NODE(5),ELEM(360,4)
      real*8 SH(5),NORMI,NORMJ,JAC,NUMER,NODE
      INTEGER ORDER,SORDR
C
C  FOR CONSTANT FIT
C
      IF(SORDR.EQ.1) THEN
        SH(1)=1.
      ELSE
C
C  CALCULATE HIGHER ORDER SHAPE FUNCTIONS
C
        DO 60 I=1,SORDR
          DENOM=1.
          NUMER=1.
          DO 55 K=1,SORDR
            IF(I.NE.K) THEN
              DENOM=(NODE(I)-NODE(K))*DENOM
              NUMER=(TERM-NODE(K))*NUMER
            END IF
 55       CONTINUE
          SH(I)=NUMER/DENOM
 60     CONTINUE
      END IF
      RETURN
      END
```

```
C*********************************************************************
C
C  SINFO
C
C  THIS SUBROUTINE READS IN THE INFORMATION NEEDED TO
C  CALCULATE THE STRUCTURE MATRICES
C
C  INPUT:
C  GE = SHEAR MODULUS
C  XNU = POISSON MODULUS
C
C*********************************************************************
      SUBROUTINE SINFO
      implicit real*8  (a-h,o-z)
      COMMON/STRUCT/GE,XNU,HS(1920,1920),GS(1920,1440),GSHAT(1920)
C
C
      READ(5,10) GE,XNU
 10   FORMAT(2F10.4)
      RETURN
      END
```

```
************************************************************************
C  SOLVE
C
C  THIS SUBROUTINE SOLVES THE LINEAR SYSTEM OF EQUATIONS
C  BY THE GAUSS ELIMINATION METHOD PROVIDING FOR
C  INTERCHANGING ROWS WHEN ENCOUNTERING A ZERO DIAGONAL
C  COEFFICIENT
C
C  A: SYSTEM MATRIX
C  B: ORIGINALLY THE INDEPENDENT COEFFICIENTS (RHS)
C     AFTER SOLUTION IT CONTAINS THE VALUES OF
C     SYSTEM UNKNOWNS
C  N: ACTUAL NUMBER OF UNKNOWNS
C  D: DETERMINANT
C
C***********************************************************************
      SUBROUTINE SOLVE(A,B,D,N,NDIM)
      implicit real*8  (a-h,o-z)
      REAL*8  A(NDIM,NDIM),B(NDIM)
      N1=N-1
      DO 100 K=1,N1
        K1=K+1
        C=A(K,K)
        IF(ABS(C)-0.000001d0) 1,1,3
 1      DO 7 J=K1,N
C
C TRY TO INTERCHANGE ROWS TO GET NON ZERO DIAGAONAL
C
          IF(ABS(A(J,K))-0.000001d0) 7,7,5
 5        DO 6 L=K,N
            C=A(K,L)
            A(K,L)=A(J,L)
 6        A(J,L)=C
          C=B(K)
          B(K)=B(J)
          B(J)=C
          C=A(K,K)
          GO TO 3
 7      CONTINUE
 8      WRITE(6,2) K
 2      FORMAT('*****SINGULARITY IN ROW',I5)
        D=0.
        GO TO 300
C
C DIVIDE ROW BY DIAGONAL COEFFICIENT
C
 3      C=A(K,K)
        DO 4 J=K1,N
 4      A(K,J)=A(K,J)/C
        B(K)=B(K)/C
C
C ELIMINATE UNKNOWN X(K) FROM ROW I
C
```

```
      DO 10 I=K1,N
        C=A(I,K)
        DO 9 J=K1,N
 9      A(I,J)=A(I,J)-C*A(K,J)
 10   B(I)=B(I)-C*B(K)
 100  CONTINUE
C
C COMPUTE LAST UNKNOWN
C
      IF (ABS(A(N,N))-0.000001d0) 101,101,102
 101  WRITE(6,2) N
      D=0.0
      GO TO 300
 102  B(N)=B(N)/A(N,N)
C
C APPLY BACKSUBSTITUTION TO COMPUTE REMAINING UNKNOWNS
C
 103  DO 200 L=1,N1
        K=N-L
        K1=K+1
        DO 200 J=K1,N
 200  B(K)=B(K)-A(K,J)*B(J)
C
C COMPUTE VALUE OF DETERMINANT
C
      D=1.
      DO 250 I=1,N
 250  D=D*A(I,I)
 300  RETURN
      END
```

```
C*********************************************************************
C
C  SOUT
C
C  THIS SUBROUTINE PRINTS OUT THE ELASTIC STRUCTURE MATRICES,
C  HS, GS AND GSHAT
C
C*********************************************************************
      SUBROUTINE SOUT
      implicit real*8  (a-h,o-z)
      COMMON/GEOMET/NNODES,NELEMS,ORDER,EPS,NORMI,NORMJ,JAC,NODEC,
     *NODPIN1,NODPIN2,NODE(5),ELEM(360,4)
      COMMON/STRUCT/GE,XNU,HS(1920,1920),GS(1920,1440),GSHAT(1920)
      real*8 NORMI,NORMJ,JAC,NODE
      INTEGER ORDER
C
C
      WRITE(6,1)
 1    FORMAT(//,22X,'*****STRUCTURAL MATRIX INFORMATION*****',//)
      WRITE(6,2)
 2    FORMAT(5X,'HS XX MATRIX',/)
      DO 10 I=1,NNODES*4
        WRITE(6,3) (HS(I,J),J=1,NNODES*3)
 3      FORMAT(35(13(F6.4,3X),/))
 10   CONTINUE
      WRITE(6,11)
 11   FORMAT(5X,'HS XY MATRIX',/)
      DO 13 I=1,NNODES*4
        WRITE(6,12) (HS(I,4*NNODES+J),J=1,NNODES*3)
 12     FORMAT(35(13(F6.4,3X),/))
 13   CONTINUE
      WRITE(6,120)
120   FORMAT(5X,'HS YX MATRIX',/)
      DO 150 I=1,NNODES*4
        WRITE(6,115) (HS(4*NNODES+I,J),J=1,NNODES*3)
115     FORMAT(35(13(F6.4,3X),/))
150   CONTINUE
      WRITE(6,220)
220   FORMAT(5X,'HS YY MATRIX',/)
      DO 250 I=1,NNODES*4
        WRITE(6,215) (HS(4*NNODES+I,4*NNODES+J),J=1,NNODES*3)
215     FORMAT(35(13(F6.4,3X),/))
250   CONTINUE
      WRITE(6,14)
 14   FORMAT(//,5X,'GS MATRIX',/)
      DO 20 I=1,NNODES*8
      WRITE(6,15) (GS(I,J),J=1,NNODES*6)
 15     FORMAT(70(13(F6.4,3X),/))
 20   CONTINUE
      WRITE(6,22)
 22   FORMAT(//,5X,'GSHAT MATRIX',/)
      WRITE(6,23) (GSHAT(I),I=1,8*NNODES)
 23   FORMAT(95(13(F7.4,3X),/))
      RETURN
      END
```

```
C*********************************************************************
C
C  SOUTPUT
C
C  THIS SUBROUTINE OUTPUTS THE DEFORMATION MATRIX, U
C  NOTE THAT AFTER THE SOLVE, THE VECTOR GSHAT NOW CONTAINS THE
C  SOLUTION DEFORMATIONS
C
C*********************************************************************
      SUBROUTINE SOUTPUT(ITER)
      implicit real*8  (a-h,o-z)
      COMMON/GEOMET/NNODES,NELEMS,ORDER,EPS,NORMI,NORMJ,JAC,NODEC,
     *NODPIN1,NODPIN2,NODE(5),ELEM(360,4)
      COMMON/STRUCT/GE,XNU,HS(1920,1920),GS(1920,1440),GSHAT(1920)
      real*8 NORMI,NORMJ,JAC,NODE
      INTEGER ORDER
C
C
      WRITE(6,1) GE,XNU,NODPIN1,NODPIN2,ITER-1
 1    FORMAT(//,22X,'*****DEFORMATION MATRIX*****',/,
     *10X,'SHEAR MODULUS = ',F10.4,5X,'POISSON MODULUS = ',F10.4,/,
     *10X,'PINNED NODES ARE = ',2I3/,10X,'ITERATION = ',I2,/)
      WRITE(6,2)
 2    FORMAT(//,5X,'U MATRIX',/)
      DO 10 I=1,4*NNODES
        WRITE(6,3) GSHAT(I),GSHAT(4*NNODES+I)
 3      FORMAT(F20.8,2X,F20.8)
 10   CONTINUE
      RETURN
      END
```

```
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
C
C  SURF
C
C  THIS ROUTINE GENERATES X,Y LOCATIONS OF LEFT, CENTER AND
C  RIGHT ENDS OF GEOMETRY PANELS FOR CYLINDER WITH OUTER AND
C  INNER RADIUS
C
C  INPUT:
C  NELEMS = NUMBER OF OUTSIDE PANELS
C           THE SAME NUMBER OF PANELS IS GENERATED ON INNER
C           SURFACE
C  RADOUT = OUTER RADIUS
C  RADIN  = INNER RADIUS
C
C  OUTPUT:
C  ELEM(I,1),ELEM(I,2) = X,Y OF LEFT END OF ELEMENT J
C  ELEM(I,3),ELEM(I,4) = X,Y OF RIGHT END OF ELEMENT J
C
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
      SUBROUTINE SURF
      implicit real*8  (a-h,o-z)
      COMMON/GEOMET/NNODES,NELEMS,ORDER,EPS,NORMI,NORMJ,JAC,NODEC,
     *NODPIN1,NODPIN2,NODE(5),ELEM(360,4)
      integer order
      real*8 NODE,NORMI,NORMJ,JAC
C
C
      PI=3.14159265
      READ(5,*) RADOUT,RADIN
 1    FORMAT(2F10.4)
C
C  GENERATE FLAT PANEL GEOMETRY
C
        ANGLE=2.*PI/NELEMS
        PLENGO=2.*RADOUT*dTAN(ANGLE/2.)
        PLENGI=2.*RADIN*dTAN(ANGLE/2.)
C
C  CALCULATE X,Y LOCATIONS FOR OUTER SURFACE
C
        DO 10 I=1,NELEMS
          ANG=(I-1)*ANGLE+PI
          RSIDE=SQRT((PLENGO/2.)**2+RADOUT**2)
          ELEM(I,1)=RSIDE*dCOS(ANG-(ANGLE/2.))
          ELEM(I,2)=RSIDE*dSIN(ANG-(ANGLE/2.))
          ELEM(I,3)=RSIDE*dCOS(ANG+(ANGLE/2.))
          ELEM(I,4)=RSIDE*dSIN(ANG+(ANGLE/2.))
 10     CONTINUE
C
C  CALCULATE X,Y LOCATIONS FOR INNER SURFACE
C
        DO 20 I=1,NELEMS
          ANG=(I-1)*ANGLE+PI
        RSIDE=SQRT((PLENGI/2.)**2+RADIN**2)
        ELEM(NELEMS+I,1)=RSIDE*dCOS(ANG-(ANGLE/2.))
        ELEM(NELEMS+I,2)=RSIDE*dSIN(ANG-(ANGLE/2.))
        ELEM(NELEMS+I,3)=RSIDE*dCOS(ANG+(ANGLE/2.))
        ELEM(NELEMS+I,4)=RSIDE*dSIN(ANG+(ANGLE/2.))
20      CONTINUE
      RETURN
      END
```

```
C*****************************************************************
C
C  VELOC
C
C  THIS SUBROUTINE CALCULATES THE U AND V COMPONENTS OF THE
C  PERTURBATION VELOCITY (UVEL,VVEL)
C
C*****************************************************************
      SUBROUTINE VELOC
      implicit real*8  (a-h,o-z)
      COMMON/FLAGS/IFLAG,PFLAG,ITER
      COMMON/GEOMET/NNODES,NELEMS,ORDER,EPS,NORMI,NORMJ,JAC,NODEC,
     *NODPIN1,NODPIN2,NODE(5),ELEM(360,4)
      COMMON/POSITION/XI(1080),YI(1080)
      COMMON/FLOW/H(240,240),G(240,240),Q(240),GHAT(240)
      COMMON/POTEN/VINF,ALPHA,VI,VJ,PINF,RHOINF,GAMMA,MINF2,NODEP
      COMMON/VELOCIT/UVEL(240),VVEL(240),XFORC(480),YFORC(480),
     *XOLD(480),YOLD(480)
      REAL*8 SHAP(5),ETA(12),WT(12),SHP(5)
      REAL*8 ETAS(8),WTS(8),SHP1(5),SHP2(5),SHP3(5)
      REAL*8 NODE,JAC,JAC1,JAC2,NORMI,NORMJ,GAMMA,MINF2
      INTEGER ORDER,PFLAG
      REAL*8 TERMQX(240,240),TERMQY(240,240),TERMPX(240,240),
     *  TERMPY(240,240)
C
C  ETA AND WT ARE 12TH ORDER GAUSSIAN QUADRATRUE VALUES AND WEIGHTS
C
      DATA ETA(1),ETA(2),ETA(3),ETA(4),ETA(5),ETA(6),ETA(7),
     *ETA(8),ETA(9),ETA(10),ETA(11),ETA(12)/-.98156063d0,-.90411726d0,
     *-.76990267d0,-.58731795d0,-.36783149d0,-.12523341d0,.12523341d0,
     *.36783149d0,.58731795d0,.76990267d0,.90411726d0,.98156063d0/
      DATA WT(1),WT(2),WT(3),WT(4),WT(5),WT(6),WT(7),WT(8),WT(9),
     *WT(10),WT(11),WT(12)/.0471753d0,.10693933d0,.16007833d0,
     *.20316743d0,.23349254d0,.24914705d0,.24914705d0,.23349254d0,
     *.20316743d0,.16007833d0,.10693933d0,.04717534d0/
C
C  ETAS AND WTS ARE 8TH ORDER GAUSSIAN VALUES AND WEIGHTS FOR
C  1/R SINGULAR INTEGRALS
C
      DATA ETAS(1),ETAS(2),ETAS(3),ETAS(4),ETAS(5),ETAS(6),ETAS(7),
     *ETAS(8)/-.00324250d0,.05349077d0,.17782733d0,.35071788d0,
     *.54581952d0,.73342708d0,.88498305d0,.97745438d0/
      DATA WTS(1),WTS(2),WTS(3),WTS(4),WTS(5),WTS(6),WTS(7),
     *WTS(8)/-3.93063681d0,1.73740631d0,.85763454d0,.53836077d0,
     *.35975759d0,.23726769d0,.14146228d0,.05874761d0/
      DATA PI/3.14159265d0/
C
C  INITIALIZE MATRICES
C
      DO 10 I=1,NNODES
        UVEL(I)=0.0
        VVEL(I)=0.0
        DO 10 J=1,NNODES
          TERMPX(I,J)=0.0
          TERMPY(I,J)=0.0
          TERMQX(I,J)=0.0
          TERMQY(I,J)=0.0
 10   CONTINUE
C
C  BEGIN LOOPS ON NODE I AND ELEMENT J
C  FOR EACH I ROW, PERFORM INTEGRATIONS OVER EACH ELEMENT J
C  AROUND THE STRUCTURE.  FILL IN ONE ROW AT A TIME
C
      DO 1000 I=1,NNODES
        DO 1000 J=1,NELEMS
C
C  DETERMINE IF THE INTEGRATION SHOULD BE SINGULAR
C  IE  IS THE NODE I ON THE ELEMENT J?
C  ISING = 0 MEANS NODE IS NOT ON ELEMENT
C  ISING = 1 MEANS NODE IS FIRST NODE ON ELEMENT J
C  ISING = 2 MEANS NODE IS SECOND NODE ON ELEMENT J
C  ISING = 3 ETC
C
          ISING=0
          DO 20 L=1,ORDER
            NODES=J*ORDER-(ORDER-L)
            IF(I.EQ.NODES) ISING=L
 20       CONTINUE
          IF(ISING.EQ.0) THEN
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
C  THE INTEGRATION IS NOT SINGULAR
C  PERFORM 12TH ORDER GAUSSIAN QUADRATURE
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
            DO 50 K=1,12
C
C  CALCULATE THE X AND Y COORDINATES OF THE QUADRATURE POINT
C  AND CALCULATE DISTANCE FROM NODE I TO QUAD POINT
C
              ETAA=-1.
              ETAC=1.
              ETAQ=ETA(K)
              SHAP(1)=(ETAQ-ETAC)/(ETAA-ETAC)
              SHAP(2)=(ETAQ-ETAA)/(ETAC-ETAA)
              X=SHAP(1)*ELEM(J,1)+SHAP(2)*ELEM(J,3)
              Y=SHAP(1)*ELEM(J,2)+SHAP(2)*ELEM(J,4)
              RI=XI(I)-X
              RJ=YI(I)-Y
              R=SQRT(RI*RI+RJ*RJ)
C
C  CALCULATE DQ*/DX,DQ*/DY,DPHI*/DX,DPHI*/DY
C
              CALL JACOBIAN(J,ETA(K))
              D=RI*(NORMI)+RJ*(NORMJ)
              DPSTDX=-RI/(R*R)
              DPSTDY=-RJ/(R*R)
              DQSTDX=((R*R*(NORMI)-D*2.*RI)/(R*R*R*R))
              DQSTDY=((R*R*(NORMJ)-D*2.*RJ)/(R*R*R*R))
C
```

```
C  CALCULATE DH/D   AND DG/D   MATRIX TERMS
C  ROW CORRESPONDS TO NODE I
C  COLUMN CORRESPONDS TO GLOBAL NUMBER OF ELEMENT NODES
C  I.E  NODE 1,2,3 ON ELEMENT 1 ARE 1,2,3
C       NODE 1,2,3 ON ELEMENT 2 ARE 4,5,6
C       NODE 1,2,3 ON ELEMENT 3 ARE 7,8,9
C
             CALL SHAPE(ORDER,ETA(K),SHAP)
             DO 35 L=1,ORDER
               JJ=J*ORDER-(ORDER-L)
               TERMQX(I,JJ)=TERMQX(I,JJ)+
     *           SHAP(L)*DQSTDX*JAC*WT(K)/(2.*PI)
               TERMPX(I,JJ)=TERMPX(I,JJ)+
     *           SHAP(L)*DPSTDX*JAC*WT(K)/(2.*PI)
               TERMQY(I,JJ)=TERMQY(I,JJ)+
     *           SHAP(L)*DQSTDY*JAC*WT(K)/(2.*PI)
               TERMPY(I,JJ)=TERMPY(I,JJ)+
     *           SHAP(L)*DPSTDY*JAC*WT(K)/(2.*PI)
 35          CONTINUE
 50        CONTINUE
          ELSE
C
C*******************************************************************
C  THE INTEGRAL IS SINGULAR ON ELEMENT J
C
C  THE DPHI*/D  TERMS ARE CALCULATED SEPARATELY FROM THE DQ*/ D TERMS
C  BECAUSE DPHI*/D  IS 1/R SINGULAR AND DQ*/D  IS 1/R*R SINGULAR
C
C  CALCULATE DQ*/D    TERMS FOR SINGULAR ELEMENT
C
C
C  BREAK UP INTERVAL INTO SINGULAR AND NON-SINGULAR PART
C  SINGULAR PART CENTERED AROUND SINGULARITY
C  START=START OF SINGULAR PORTION
C  END=END OF SINGULAR PORTION
C
            SING=NODE(ISING)
            DELTA=(1.-ABS(NODE(ISING)))
            START=NODE(ISING)-DELTA
            END=NODE(ISING)+DELTA
            SUM=0.0
            ETAA=-1.
            ETAC=1.
C
C  IF START OF SINGULAR INTEGRAL IS -1 AND END IS 1 THEN
C  THE ENTIRE ELEMENT IS IN THE SINGULAR INTERVAL
C  SKIP THE NON-SINGULAR CALCULATIONS
C
            IF(START.EQ.-1.d0.AND.END.EQ.1.d0) GO TO 30
            DO 250 K=1,12
C
C  START OF SINGULAR INTERVAL IS -1, END IS END
C  THERFORE NON-SINGULAR PART IS FROM END TO 1
C
```

```
                IF(START.EQ.-1.d0) THEN
                  ET=(ETA(K)+1.)*(1.-END)/2.+END
                  CALL SHAPE(ORDER,ET,SHP)
                  ETAQ=ET
                  SHAP(1)=(ETAQ-ETAC)/(ETAA-ETAC)
                  SHAP(2)=(ETAQ-ETAA)/(ETAC-ETAA)
                  X=SHAP(1)*ELEM(J,1)+SHAP(2)*ELEM(J,3)
                  Y=SHAP(1)*ELEM(J,2)+SHAP(2)*ELEM(J,4)
                  RI=XI(I)-X
                  RJ=YI(I)-Y
                  R=SQRT(RI*RI+RJ*RJ)
                  CALL JACOBIAN(J,ET)
                  DQSTDX=(NORMI)/(R*R)
                  DQSTDY=(NORMJ)/(R*R)
                  DO 100 L=1,ORDER
                    JJJ=J*ORDER-(ORDER-L)
                    TERMQX(I,JJJ)=TERMQX(I,JJJ)+
     *              SHP(L)*DQSTDX*JAC*.5*(1.-END)*WT(K)/(2.*PI)
                    TERMQY(I,JJJ)=TERMQY(I,JJJ)+
     *              SHP(L)*DQSTDY*JAC*.5*(1.-END)*WT(K)/(2.*PI)
100               CONTINUE
                ELSE
C
C  END OF SINGULAR INTERVAL IS 1
C  THEREFORE NON-SINGULAR PART IS FROM -1 TO START
C
                  ET=(ETA(K)+1.)*(START+1.)/2.-1.
                  CALL SHAPE(ORDER,ET,SHP)
                  ETAQ=ET
                  SHAP(1)=(ETAQ-ETAC)/(ETAA-ETAC)
                  SHAP(2)=(ETAQ-ETAA)/(ETAC-ETAA)
                  X=SHAP(1)*ELEM(J,1)+SHAP(2)*ELEM(J,3)
                  Y=SHAP(1)*ELEM(J,2)+SHAP(2)*ELEM(J,4)
                  RI=XI(I)-X
                  RJ=YI(I)-Y
                  R=SQRT(RI*RI+RJ*RJ)
                  CALL JACOBIAN(J,ET)
                  DQSTDX=(NORMI)/(R*R)
                  DQSTDY=(NORMJ)/(R*R)
                  DO 200 L=1,ORDER
                    JJJ=J*ORDER-(ORDER-L)
                    TERMQX(I,JJJ)=TERMQX(I,JJJ)+
     *              SHP(L)*DQSTDX*JAC*.5*(START+1.)*WT(K)/(2.*PI)
                    TERMQY(I,JJJ)=TERMQY(I,JJJ)+
     *              SHP(L)*DQSTDY*JAC*.5*(START+1.)*WT(K)/(2.*PI)
200               CONTINUE
                END IF
250           CONTINUE
C
C  NOW CALCULATE SINGULAR INTEGRAL
C
300           CONTINUE
              DO 400 K=1,12
                X2X1=ELEM(J,3)-ELEM(J,1)
                Y2Y1=ELEM(J,4)-ELEM(J,2)
                PLENG=SQRT(X2X1*X2X1+Y2Y1*Y2Y1)
                ET=.5*(END-START)*(ETA(K)+1.)+START
                CALL JACOBIAN(J,ET)
                DQSTDX=NORMI*8./(PLENG*PLENG*(END-START))
                DQSTDY=NORMJ*8./(PLENG*PLENG*(END-START))
                CALL SHAPE(ORDER,ET,SHP)
                DO 400 L=1,ORDER
                  JJJ=J*ORDER-(ORDER-L)
                  TERMQX(I,JJJ)=TERMQX(I,JJJ)+
     *            SHP(L)*DQSTDX*JAC*WT(K)/(ETA(K)**2*(2.*PI))
                  TERMQY(I,JJJ)=TERMQY(I,JJJ)+
     *            SHP(L)*DQSTDY*JAC*WT(K)/(ETA(K)**2*(2.*PI))
400           CONTINUE
C
C  CALCULATE SUMMATION OF WT/ETA2
           DO 500 K=1,12
             SUM=SUM+WT(K)/ETA(K)**2
500        CONTINUE
C
C  CALCULATE F(0) AND ADD -(SUM+2)F(0) TO TERMS
C
           ET=.5*(END-START)+START
           CALL JACOBIAN(J,ET)
           DQSTDX=NORMI*8./(PLENG*PLENG*(END-START))
           DQSTDY=NORMJ*8./(PLENG*PLENG*(END-START))
           CALL SHAPE(ORDER,ET,SHP)
           DO 600 L=1,ORDER
             JJJ=J*ORDER-(ORDER-L)
             TERMQX(I,JJJ)=TERMQX(I,JJJ)-
     *       (SUM+2.)*SHP(L)*DQSTDX*JAC/(2.*PI)
             TERMQY(I,JJJ)=TERMQY(I,JJJ)-
     *       (SUM+2.)*SHP(L)*DQSTDY*JAC/(2.*PI)
600        CONTINUE
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
C  CALCULATE DPHI*/D  TERMS FOR THE SINGULAR ELEMENT
C
           X2X1=ELEM(J,3)-ELEM(J,1)
           Y2Y1=ELEM(J,4)-ELEM(J,2)
           PLENG=SQRT(X2X1*X2X1+Y2Y1*Y2Y1)
           PX=X2X1*(2./PLENG**2)*(1./(2.*PI))
           PY=Y2Y1*(2./PLENG**2)*(1./(2.*PI))
C
C  PERFORM 8TH ORDER SUMMATION
C
           DO 740 K=1,8
             ET=(-1.-SING)*ETAS(K)+SING
             CALL JACOBIAN(J,ET)
             JAC1=JAC
             CALL SHAPE(ORDER,ET,SHP1)
             ET=(1.-SING)*ETAS(K)+SING
             CALL JACOBIAN(J,ET)
             JAC2=JAC
             CALL SHAPE(ORDER,ET,SHP2)
```

```
                DO 730 L=1,ORDER
                  JJJ=J*ORDER-(ORDER-L)
                  TERMPX(I,JJJ)=TERMPX(I,JJJ)-
     *            SHP1(L)*PX*JAC1*WTS(K)+SHP2(L)*PX*JAC2*WTS(K)
                  TERMPY(I,JJJ)=TERMPY(I,JJJ)-
     *            SHP1(L)*PY*JAC1*WTS(K)+SHP2(L)*PY*JAC2*WTS(K)
 730            CONTINUE
 740          CONTINUE
C
C  CALCULATE F(SINGULARITY) LOG TERMS AND ADD TO THE SUMMATION
C
              CALL SHAPE(ORDER,SING,SHP3)
              CALL JACOBIAN(J,SING)
              DO 750 L=1,ORDER
                JJJ=J*ORDER-(ORDER-L)
                TERMPX(I,JJJ)=TERMPX(I,JJJ)-
     *          SHP3(L)*PX*JAC*dlog(ABS(-1.-SING))+
     *          SHP3(L)*PX*JAC*dlog(ABS(1.-SING))
                TERMPY(I,JJJ)=TERMPY(I,JJJ)-
     *          SHP3(L)*PY*JAC*dlog(ABS(-1.-SING))+
     *          SHP3(L)*PY*JAC*dlog(ABS(1.-SING))
 750          CONTINUE
            END IF
 1000 CONTINUE
C
C  IF PRINT FLAG = 1 PRINT OUT TERM MATRICES
C
      IF(PFLAG.EQ.1.OR.PFLAG.EQ.3) CALL VELOUT(TERMQX,
     *          TERMQY,TERMPX,TERMPY)
C
C  CALCULATE U AND V PERTURBATION VELOCITIES
C
      DO 1200 I=1,NNODES
        DO 1200 J=1,NNODES
          UVEL(I)=UVEL(I)+(-TERMQX(I,J)*GHAT(J)
     *      +TERMPX(I,J)*Q(J))
          VVEL(I)=VVEL(I)+(-TERMQY(I,J)*GHAT(J)
     *      +TERMPY(I,J)*Q(J))
 1200 CONTINUE
      DO 1250 I=1,NNODES
        UVEL(I)=UVEL(I)/.5
        VVEL(I)=VVEL(I)/.5
1250  CONTINUE
      do 1300 i=1,nnodes
        uvel(i)=vi-uvel(i)
        vvel(i)=vj-vvel(i)
 1300 continue
      RETURN
      END
```

```
C**********************************************************************
C
C  VELOUT
C
C  THIS SUBROUTINE PRINTS THE INTERMEDIATE PARTS OF THE
C  VELOCITY MATRICES  FOR CHECKOUT
C
C**********************************************************************
      SUBROUTINE VELOUT(TERMQX,TERMQY,TERMPX,TERMPY)
      implicit real*8  (a-h,o-z)
      COMMON/GEOMET/NNODES,NELEMS,ORDER,EPS,NORMI,NORMJ,JAC,NODEC,
     *NODPIN1,NODPIN2,NODE(5),ELEM(360,4)
      real*8 TERMQX(240,240),TERMQY(240,240),TERMPX(240,240),
     *TERMPY(240,240)
      real*8 NORMI,NORMJ,JAC,NODE
      INTEGER ORDER
C
C
      WRITE(6,1)
 1    FORMAT(//,5X,'TERMQX MATRIX',/)
      DO 10 I=1,NNODES
        WRITE(6,5) (TERMQX(I,J),J=1,NNODES)
 5      FORMAT(/13(8(F10.4,5X),/))
 10   CONTINUE
      WRITE(6,11)
 11   FORMAT(//,5X,'TERMQY MATRIX',/)
      DO 20 I=1,NNODES
        WRITE(6,15) (TERMQY(I,J),J=1,NNODES)
 15     FORMAT(/13(8(F10.4,5X),/))
 20   CONTINUE
      WRITE(6,21)
 21   FORMAT(//,5X,'TERMPX MATRIX',/)
      DO 30 I=1,NNODES
        WRITE(6,25) (TERMPX(I,J),J=1,NNODES)
 25     FORMAT(/13(8(F10.4,5X),/))
 30   CONTINUE
      WRITE(6,31)
 31   FORMAT(//,5X,'TERMPY MATRIX',/)
      DO 40 I=1,NNODES
        WRITE(6,35) (TERMPY(I,J),J=1,NNODES)
 35     FORMAT(/13(8(F10.4,5X),/))
 40   CONTINUE
      RETURN
      END
```

Appendix C: Background Concepts

Weighted Residuals

The method of weighted residuals may be defined in the following manner. Begin with the problem definition;

$$L(u_0) = b \quad \text{in domain } \Omega \tag{C1}$$

with boundary conditions

$$S(u_0) = s \quad \text{on boundary } \Gamma_1 \tag{C2}$$

$$G(u_0) = g \text{ on boundary } \Gamma_2$$

where u_0 is the exact solution to the problem. The approximation may be made so that

$$u_0 \approx u = \sum_{k=1}^{n} a_k N_k + a_o \tag{C3}$$

where a_k are undetermined coefficients and N_k are linearly independent functions. Substituting the approximation into Equations C1 and C2:

$$\begin{aligned} L(u) - b &\neq 0 = R \\ S(u) - s &\neq 0 = R_1 \\ G(u) - g &\neq 0 = R_2 \end{aligned} \tag{C4}$$

where R, R_1 and R_2 are the residuals, or the errors. The purpose of the various numerical methods techniques is to make all the errors as small as possible over the domain and boundary. One way of doing this is with the weighted residual method.

Take a function w, such that

$$w = \sum_{k=1}^{m} b_k \psi_k \tag{C5}$$

The error can be distributed throughout domain Ω by multiplying the error by the function w which is called a weighting function.

$$\int_{\Omega} R\, w\, d\Omega = 0 \tag{C 6}$$

Because the b terms are arbitrary, the key is to choose the function ψ_k so that

$$\int_{\Omega} R\, \psi_k\, d\Omega = 0 \tag{C7}$$

Shape (Interpolation) Functions

The value of a function u may be defined in terms of its value at known points (u_k) and shape, or interpolation, functions (N_k) such that;

$$\begin{aligned} u(\eta) &= N_1(\eta)\, u_1 + N_2(\eta)\, u_2 + \ldots + N_k(\eta)\, u_k \\ &= \begin{bmatrix} N_1(\eta) & N_2(\eta) & \ldots & N_k(\eta) \end{bmatrix} \begin{Bmatrix} u_1 \\ u_2 \\ \ldots \\ u_k \end{Bmatrix} \end{aligned} \tag{C8}$$

k is the order of the function such as 1st (constant), 2nd (linear), 3rd (parbolic), 4th, etc. The points at which the value of u is defined are called the nodal points, or nodes.

Lagrangian polynomial shape functions were used in the development of this thesis are may be calculated by:

$$\Phi_1(\eta) = \frac{(\eta - \eta_1) \ldots (\eta - \eta_{i-1})\, (\eta - \eta_{i+1}) \ldots (\eta - \eta_k)}{(\eta_i - \eta_1) \ldots (\eta_i - \eta_{i-1})\, (\eta_i - \eta_{i+1}) \ldots (\eta_i - \eta_k)} \tag{C9}$$

The variable η is an elemental coordinate. η varies from -1 at the left end to +1 at the right end of the element. The nodes are therefore distributed throughout the element with values from -1 to 1. For example, a third order function may have nodes defined at -1, 0, +1.

Jacobians

A Jacobian allows the transformation from the integral about the boundary ($d\Gamma$) to the integral along the element surface ($d\eta$). For a two-dimensional problem the Jacobian is defined as;

$$|J| = \sqrt{\left(\frac{\partial x}{\partial \eta}\right)^2 + \left(\frac{\partial y}{\partial \eta}\right)^2} = \frac{\partial \Gamma}{\partial \eta} \tag{C10}$$

Therefore,

$$d\Gamma = |J|\ d\eta \tag{C11}$$

In order to calculate the Jacobian, x and y must be expressed in terms of the variable η. For a flat element where (x_1, y_1) are the coordinated for the left-end of the element and (x_2, y_2) are the coordinates for the right-end, x and y may be expressed by shape functions;

$$x = N_1 x_1 + N_2 x_2 \qquad y = N_1 y_1 + N_2 y_2 \tag{C12}$$

and

$$\frac{\partial x}{\partial \eta} = \frac{\partial \Phi_1}{\partial \eta} x_1 + \frac{\partial \Phi_2}{\partial \eta} x_2 \qquad \frac{\partial y}{\partial \eta} = \frac{\partial \Phi_1}{\partial \eta} y_1 + \frac{\partial \Phi_2}{\partial \eta} y_2 \tag{C13}$$

Bibliography

1. Bisplinghoft, R.L. and Ashley, H. Principles of Aeroelasticity. New York: Dover Publications, Inc, 1975.

2. Brebbia, C.A. The Boundary Element Method for Engineers (Third Revised Edition). Plymouth, England: Penteck Press, 1984.

3. Brebbia, C.A. and others. Boundary Element Techniques, Theory and Applications in Engineering. Berlin: Springer-Verlay, 1984.

4. Hernandez, E. "MILSTAR/FSED Airworthiness Flight Test Plan", 4950TW-FTP-88-02-01, 4950th Test Wing, Wright-Patterson AFB, Ohio, May 1988.

5. Hess, J.L. and Smith, A.M.O. "Calculation of Potential Flow About Arbitrary Bodies," Progres in Aeronautical Sciences, Vol 8., edited by D. Kuchemann. London: Pergamon Press, 1967.

6. Kawakami, T. and Kitahara, M. "Analysis of Structure-Fluid Dynamic Interaction Problems by Boundary Integral Equation Methods", Boundary Elements VII Conference, 515-524. 1986.

7. Kellogg, O.D. Foundations of Potential Theory. New York: Dover, 1953.

8. Landahl, M. "Pressure-Loading Functions for Oscillating Wings With Control Surfaces", AIAA Journel, Vol 6, No 2, Feb 1968.

9. van Nieker, Becker. " *Integration of Singular Functions* Associated with Lifting Surface Theory," AIAA Journal, Technical Notes, Vol 24, No 7, 1194-1195, July 1986.

10. Youngren, Harold and others. "Quadrilateral Element Panel Method (QUADPAN)," User's Manual, Ver 3.2, Lockheed-California Company, 1984.

11. Zienkiewitz, O.C. and Bettess, P. "Fluid-Structure Dynamic Interaction and Wave Forces, An Introduction to Numerical Treatment," International Journal for Numerical Methods Engineering, 1-16, Vol 13, 1978.

VITA

Norma F. Taylor [redacted] Graduating from Morristown High School, Morristown Indiana in 1977, she then attended Purdue University and received the degree of Bachelor of Science in Aeronautical and Astronautical Engineering in May 1982. Upon graduation she was employed by the Flight Dynamics Laboratory, Wright-Patterson AFB, Ohio, as an engineer on the X-29 Forward Swept Wing motion simulation team. Since October 1983 she has been a flight test engineer at the 4950th Test Wing, Wright-Patterson AFB, Ohio , performing airworthiness flight tests on the Wing's modified aircraft. She entered the School of Engineering, Air Force Institute of Technology June 1986.

[redacted]

[redacted]

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) AFIT/GAE/AA/88D-37	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION School of Engineering 6b. OFFICE SYMBOL (If applicable) AFIT/AA	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS (City, State, and ZIP Code) Air Force Institute of Technology (AU) Wright-Patterson AFB OH 45433-6583	7b. ADDRESS (City, State, and ZIP Code)
8a. NAME OF FUNDING/SPONSORING ORGANIZATION 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code)	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO / WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)
SOLUTION OF POTENTIAL FLOW PAST AN ELASTIC BODY USING THE BOUNDARY ELEMENT TECHNIQUE

12. PERSONAL AUTHOR(S)
Norma Faye Taylor, B.S.

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
MS Thesis	FROM ______ TO ______	1988 December	128

16. SUPPLEMENTARY NOTATION

17. COSATI CODES

FIELD	GROUP	SUB-GROUP
20	04	
20	11	

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
Aeroelasticity, Elastic Deformation, Potential Flow, Boundary Element Method, Boundary Integral, Numerical Methods and Procedures

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

Thesis Chairman: Lanson J. Hudson, Major, USAF
Instructor of Aeronautical Engineering

[illegible]
12 Jan 1989

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT [✓ UNCLASSIFIED/UNLIMITED] ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL Lanson J. Hudson, Major, USAF	22b. TELEPHONE (Include Area Code) (513) 255-3708 22c. OFFICE SYMBOL AFIT/AA

DD Form 1473, JUN 86 Previous editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

Block 19. ABSTRACT

This thesis describes the development of a Fortran computer code which models the interaction between an incompressible, potential flow and a homogeneous, elastic structure. The boundary element technique was chosen over finite differences and finite elements because of its ability to numerically approximate both the fluid and structural behavior with a common definition of the fluid/structure boundary.

The ability to accurately model solid and fluid boundaries can be quite important in the fields of aeroelasticity and structural analysis. The nature of these boundaries is what determines the final solution to a problem of fluid flow past an elastic body. Often the complexity of defining and tracking the boundary and its associated boundary conditions has led the user to assumptions of rigid bodies, and therefore rigid boundaries. Certainly the tasks of defining the domain grids for finite difference and finite element techniques have not simplified this process.

In the computer code developed for this thesis the fluid and structural governing equations are simultaneously solved to determine the pressure about the structure and the corresponding elastic deformations. The deformations are applied to the original boundary, resulting in a new geometry. This new geometry is used to recalculate the pressure field about the structure, and the process is iterated until a final steady-state solution is obtained.

While simplifying assumptions have been made in the execution of this thesis, the general boundary element technique has the ability to model complex higher order problems. The initial results obtained during the course of this work show promise, and follow-on studies are recommended.